TYPE Review
PUBLISHED 26 February 2024
DOI 10.3389/fdata.2024.1304439

Check for updates

OPEN ACCESS

EDITED BY
Emanuel Lacić,
Infobip d.o.o., Croatia

REVIEWED BY
Bettina Fazzinga,
University of Calabria, Italy
Simone Kopeinik,
Know Center, Austria

*CORRESPONDENCE
Mathias Uta
✉ mathias.uta@siemens-energy.com

RECEIVED 29 September 2023
ACCEPTED 06 February 2024
PUBLISHED 26 February 2024

CITATION
Uta M, Felfernig A, Le V-M, Tran TNT, Garber D, Lubos S and Burgstaller T (2024) Knowledge-based recommender systems: overview and research directions.
Front. Big Data 7:1304439.
doi: 10.3389/fdata.2024.1304439

Knowledge-based recommender systems: overview and research directions

Mathias Uta[1]*, Alexander Felfernig[2], Viet-Man Le[2], Thi Ngoc Trang Tran[2], Damian Garber[2], Sebastian Lubos[2] and Tamim Burgstaller[2]

[1]Siemens Energy AG, Erlangen, Germany, [2]Institute of Software Technology (IST) - Applied Software Engineering & Ai Research Group (ASE), Graz University of Technology, Graz, Austria

Recommender systems are decision support systems that help users to identify items of relevance from a potentially large set of alternatives. In contrast to the mainstream recommendation approaches of collaborative filtering and content-based filtering, knowledge-based recommenders exploit semantic user preference knowledge, item knowledge, and recommendation knowledge, to identify user-relevant items which is of specific relevance when dealing with complex and high-involvement items. Such recommenders are primarily applied in scenarios where users specify (and revise) their preferences, and related recommendations are determined on the basis of constraints or attribute-level similarity metrics. In this article, we provide an overview of the existing state-of-the-art in knowledge-based recommender systems. Different related recommendation techniques are explained on the basis of a working example from the domain of survey software services. On the basis of our analysis, we outline different directions for future research.

KEYWORDS

recommender systems, semantic recommender systems, knowledge-based recommender systems, case-based recommendation, constraint-based recommendation, critiquing-based recommendation, constraint solving, model-based diagnosis

1 Introduction

Recommender systems support users in identifying relevant items from a large set of alternatives and thus help to reduce the complexity of decisions and increase user satisfaction and sales (Felfernig and Burke, 2008; Burke et al., 2011; Knijnenburg et al., 2012). *Single-shot recommender systems* recommend items based on already stored preferences of a user (e.g., videos a user purchased/liked in the past) without the need of engaging the user in a dialog for capturing further preferences (Rafter and Smyth, 2005; Pramod and Bafna, 2022). In contrast, *conversational recommender systems* perform preference elicitation on the basis of a user/recommender dialog (Goeker and Thompson, 2000; Bridge, 2002; Gao et al., 2021). Depending on the application scenario, different recommendation approaches can be applied (see Tables 1, 2 for an overview of used data sources and properties of those recommendation approaches).

First, collaborative filtering (CF) (Ekstrand et al., 2011) is based on the idea of simulating word-of-mouth promotion by determining recommendations on the basis of the preferences of so-called nearest neighbors (NNs), which are users with preferences similar to those of the current user. For example, a movie that has been watched by the nearest neighbors of the current user (but not by the current user) and has been evaluated positively by those nearest neighbors should also be recommended to the current user. A major advantage of CF approaches is an easy setup since no detailed information about the provided items is needed. Furthermore, serendipity effects can be created, i.e., collaborative filtering is a good choice for finding recommendations which are (in a positive sense) a kind of "surprise" for the user, thus supporting diversity and the identification of completely unexpected/unsearched items (Iaquinta et al., 2008; Ziarani and Ravanmehr, 2021). CF typically follows the idea of single-shot recommendation where no session-specific user dialogs are needed. A disadvantage of CF recommenders is cold-start problems which are due to the sparsity of rating information (regarding users and items) available in the recommendation algorithm, making it hard to identify accurate recommendations (Lika et al., 2014).

Second, content-based filtering (CBF) (Pazzani and Billsus, 2007) is based on the idea of recommending items which are in one way or another similar to items the user has consumed in the past. In contrast to CF, CBF does not exploit the preferences of nearest neighbors which limits the cold-start problem to the availability of initial user item ratings. A kind of disadvantage of CBF recommendation approaches is need of item knowledge, for example, in terms of item categories and/or item descriptions which are used to derive a user profile representing a user item preference history. CBF systems are typically of type "single-shot", i.e., no conversational user interface needs to be provided for preference elicitation purposes. *Basic CBF* that purely focuses on the recommendation of similar items is less appropriate when it comes to the triggering of serendipity effects (Iaquinta et al., 2008). For example, if a user has purchased the song *Personal Jesus* from *Depeche Mode*, an item that could be recommended in the future could be the same song performed by *Johnny Cash*. However, further developments in content-based recommendation have improved the capability of CBF with regard to the achievement of serendipity effects, for example, on the basis of increasing the diversity of recommended items (Maccatrozzo et al., 2017). In the line with CF approaches, CBF has an easy setup process, since only item knowledge is needed in the setup phase. Since preferences change over time, both CF and CBF have an issue with item domains characterized by infrequently articulated and changing preferences. For example, time intervals between individual car purchases of a user could be around 10 years—in such long time periods, user preferences could completely change, which makes the identification of recommendations challenging without using a preference elicitation dialog.

Third, knowledge-based recommender (KBR) systems can be considered as complementary to CF- and CBF-based approaches in terms of avoiding the related cold-start difficulties (Burke, 2000; Towle and Quinn, 2000; Lorenzi and Ricci, 2005). KBR systems are based on the idea of collecting the preferences of a user (preference elicitation) within the scope of a dialog and then to recommend items either (1) on the basis of a predefined set of recommendation rules (constraints) or (2) using similarity metrics that help to identify items which are similar to the preferences of the user. The first approach is denoted as *constraint-based recommendation* (Felfernig and Burke, 2008), whereas the second one is referred to as *case-based recommendation* (Lorenzi and Ricci, 2005)—these two can be regarded as major types of knowledge-based recommender systems (Aggarwal, 2016). Serendipity effects in knowledge-based recommendation are limited by the static encoding in terms of constraints (rules) and similarity metrics. KBR systems support the determination of recommendations specifically in complex and high-involvement item domains [domains where suboptimal decisions can have significant negative consequences, for example, when investing in high-risk financial services (Felfernig et al., 2006)] where items are not bought on a regular basis (Aggarwal, 2016). Example item domains are financial services (Felfernig et al., 2007; Musto et al., 2015), software services (Felfernig et al., 2021), apartment or house purchasing (Fano and Kurth, 2003), and digital cameras (Felfernig et al., 2006). These systems are able to take into account constraints (e.g., high-risk financial services must not be recommended to users with a low preparedness to take risks) and provide explanations of recommendations also in situations where no solution could be identified. In contrast to CF and CBF, KBR systems support explicit preference elicitation dialogs which makes them immune with regard to user preferences changing over time. Due to an often time-intensive knowledge exchange between domain experts and knowledge engineers, the definition of recommendation knowledge can trigger high setup costs (Ulz et al., 2017). Both types of KBR systems (case-based and constraint-based) are conversational, since a preference elicitation dialog helps to figure out user preferences (Gao et al., 2021).

1.1 Further recommendation approaches

In addition to the three basic approaches of collaborative filtering, content-based filtering, and knowledge-based recommendation, there exist various further approaches. First, *hybrid recommender systems* (Burke, 2002) exploit synergy effects by combining different recommendation approaches. For example, by combining CBF with CF, the cold-start problem can be solved by initially applying CBF in order to collect the relevant rating data needed for establishing a CF-based approach. Furthermore, *group recommender systems* (Masthoff, 2015; Dara et al., 2020; Felfernig et al., 2024b) support the determination of recommendations for whole groups, i.e., not individual users. These systems apply basic recommendation approaches in such a way that recommendations satisfy the preferences of all or at least a subset of the group members. For example, the *average* user rating per item can be regarded as an item rating provided by the whole group.

1.2 Article contributions

The major contributions of this article are as follows. First, we provide an overview of the existing state-of-the-art in knowledge-based recommender systems. Our work significantly enhances existing overviews on the same topic (see the last row of Table 3) specifically in terms of the inclusion of new technological approaches and a broader view on how knowledge-based technologies are applied to recommender systems. Second, to assure understandability, we explain different underlying recommendation techniques on the basis of a working example. Third, to foster further related work, we discuss different open research directions derived from our literature analysis.

TABLE 1 Basic recommendation approaches of collaborative filtering (CF), content-based filtering (CBF), and knowledge-based recommendation (KBR), and used data sources including the current user preferences, a user's preference history, the preferences of nearest neighbors, and item knowledge.

Approach	Preferences	Preference history	Nearest neighbors	Item knowledge
Collaborative (CF)	–	×	×	–
Content-based (CBF)	–	×	–	×
Knowledge-based (KBR)	×	–	–	×

TABLE 2 *Basic properties* of different recommendation approaches: *easy setup* = low effort needed for setting up the recommender system, *dialog-based* = conversational process between system and user, *serendipity* = effect of proposing unexpected but relevant recommendations, *cold-start problem* = initial data are needed to provide reasonable recommendations, *high-involvement items* = a user carefully evaluates the candidate items since suboptimal decisions can have significant negative consequences.

Recommendation approach	Collaborative (CF)	Content-based (CBF)	Knowledge-based (KBR)
Easy setup	×	×	–
Dialog-based	–	–	×
Serendipity	×	–	×
Cold-start problem	×	×	–
High-involvement items	–	–	×

The remainder of this article is organized as follows. In Section 2, we explain the methodology applied for our literature analysis. An introduction to basic knowledge-based recommendation approaches is presented in Section 3, where we specifically focus on techniques and applications of case-based and constraint-based recommendation. Thereafter, in Section 4, we provide an overview of advanced techniques in knowledge-based recommendation with a specific focus on integration scenarios with machine learning. In this context, we also discuss related topics such as hybrid recommendation with knowledge-based recommenders and knowledge-based group recommender systems. In Section 5, we discuss the identified research directions. The article is concluded with Section 6.

2 Methodology

Our overview of the existing state-of-the-art in knowledge-based recommendation (KBR) is based on a literature analysis conducted using the activities of *search* (querying of leading research portals), *review* (evaluating and classifying the identified scientific contributions), and *discussion of reviewed contributions* (deciding about inclusion of the identified studies in our overview based on the selection criteria of quality and relevance). Queries have been performed between December 2023 and January 2024 on the research platforms Google Scholar,[1] ResearchGate,[2] ScienceDirect,[3] SpringerLink,[4] and Elsevier,[5] using the keywords of "recommender systems" + ["knowledge-based" | "case-based" | "critiquing" | "ontologies" | "knowledge graphs" | "constraint satisfaction" | "conversational" | "mass customization" | "groups" | "overview"], representing 10 individual search queries. Using these queries, we have also analyzed topic-related scientific contribution in conferences and journals: the International Joint Conference on Artificial Intelligence (IJCAI), the AAAI Conference on Artificial Intelligence, the European Conference on Artificial Intelligence (ECAI), the Recommender Systems Conference (RecSys), the User Modeling, Adaptation, and Personalization (UMAP) conference, the Constraint Programming Conference (CP), the Software Product Line Conference (SPLC), User Modeling and User-Adapted Interaction (UMUAI), and ACM Transactions on Recommender Systems. Thereafter, using the snowballing technique (Wohlin, 2014), we have analyzed the reference sections of the identified studies. Major criteria for estimating the relevance of a paper were (1) a *topic-wise match*, i.e., a relationship to knowledge-based recommenders and (2) an *official publication* in a workshop, conference, journal, magazine, book, or PhD thesis. With our analysis, we have identified 97 publications directly related to the topic of knowledge-based recommendation that have been used as a basis for this article (see Table 3).

1 https://scholar.google.com/
2 https://www.researchgate.net/
3 https://www.sciencedirect.com/
4 https://link.springer.com/
5 https://www.elsevier.com/

3 Basic approaches and applications

In this section, we introduce a working example from the domain of *survey software service recommendation* that is used to explain different knowledge-based recommendation approaches. The underlying scenario is that users search for a survey software configuration that fulfills their preferences, for example, a researcher received a new project funding and is now interested in purchasing a survey software that supports different user studies envisioned for the new project. In many cases, Web applications (services) are preferred over solutions requiring an installation at the customer site. In such a scenario, knowledge-based recommenders can support users in identifying an appropriate

TABLE 3 Overview of scientific contributions in knowledge-based recommender systems organized in the line of the primary focus of the paper: ALG, algorithms; PREF, preference elicitation; KA, knowledge acquisition; APP, application; FUR, further recommendation approaches and knowledge representations; OV, overview articles/papers including aspects related to knowledge-based recommendation.

Approach	ALG	PREF	KA	APP
Case-based (CBR)	Jannach, 2006	Goeker and Thompson, 2000; Bridge, 2002; McSherry, 2003; Mirzadeh et al., 2005; Christakopoulou et al., 2016	Khan and Hoffmann, 2003; Zou et al., 2020	Fesenmaier et al., 2003; Lee and Kim, 2015; Musto et al., 2015; Feely et al., 2020; Bokolo, 2021; Hernandez-Nieves et al., 2021
Critiquing-based (CRIT)	Reilly et al., 2004; Smyth et al., 2004; McCarthy et al., 2005; Mandl and Felfernig, 2012; Murti et al., 2016	Zhang et al., 2008; Chen et al., 2017; Xie et al., 2018; Wu et al., 2019; Güell et al., 2020	McCarthy et al., 2010	Burke et al., 1996; Grasch et al., 2013
Constraint-based (CON)	Cöster et al., 2002; Felfernig and Burke, 2008; Falkner et al., 2011; Fargier et al., 2016; Erdeniz et al., 2019; Teppan and Zanker, 2020; Felfernig et al., 2023a	Towle and Quinn, 2000; Fano and Kurth, 2003; Junker, 2004; Felfernig et al., 2009b, 2012, 2013c, 2018a,b; Pereira et al., 2016; Tazl et al., 2019; Erdeniz et al., 2022; Uta et al., 2022	Jannach and Kreutler, 2007; Felfernig et al., 2009a, 2013a,b, 2015; Daoudi et al., 2016; Uta et al., 2021; Lubos et al., 2023	Felfernig and Kiener, 2005; Felfernig et al., 2006, 2007; Zanker et al., 2006; Murphy et al., 2015; Wobcke et al., 2015; Ulz et al., 2017; Almalis et al., 2018
Further (FUR)	McCarthy et al., 2006	Burke, 2002; Zhou et al., 2020; Zhu et al., 2020; Le et al., 2022	Wang et al., 2019a,b; Sun et al., 2020; Esheiba et al., 2021; Sha et al., 2021	Lee et al., 2006; Bahramian and Ali Abbaspour, 2015; Colombo-Mendoza et al., 2015; Pessemier et al., 2017; Cordero et al., 2020; Dong et al., 2020
Overviews (OV)	Bridge et al., 2005; Felfernig et al., 2011, 2024b; Aggarwal, 2016; Gao et al., 2021	Lorenzi and Ricci, 2005; Chen and Pu, 2012; Masthoff, 2015; Atas et al., 2021; Felfernig et al., 2021; Jannach et al., 2021; Tran et al., 2023	Felfernig, 2007; Bouraga et al., 2014; Felfernig et al., 2014; Cena et al., 2021; Popescu et al., 2022	Burke, 2000; Felfernig et al., 2024a

configuration (including pricing) of a survey software service that fulfills their wishes and needs. In our example, we include the user-selectable survey software features *ABtesting* (should ABtesting be supported by the survey software), *statistics* (should a basic statistics feature be included), *multiplechoice* (should the specification of multiple-choice questions be possible), and *license* [which license model is preferred—*free of charge* (0) vs. *with costs*, i.e., 100]. In the following, we show how knowledge-based recommenders can support the identification of relevant items, i.e., survey software configurations.

Knowledge-based recommender systems can operate on different knowledge representations—these knowledge representations will be explained and exemplified in Section 3.1. Thereafter, we introduce the two basic approaches to knowledge-based recommendation, which are (1) *case-based recommendation* including *critiquing-based recommendation* as a specific form of case-based recommendation (see Section 3.2) and (2) *constraint-based recommendation* (see Section 3.3).

3.1 Recommendation knowledge representations

Knowledge representations of knowledge-based recommender systems can be (1) *table-based* which is used in scenarios where items are represented in terms of product table entries or (2) *constraint-based* which is used in scenarios where items are defined on the basis of a set of *restrictions* (also denoted as *rules* or *constraints*). In the first case (extensional representation—see, for example, Table 4), each item that could be recommended is explicitly defined in a corresponding item (product) table. In the second case, there is no need to enumerate all items since items are specified in a constraint-based fashion (intensional representation—see, for example, Table 5).

3.1.1 Table-based representations

In many knowledge-based recommendation scenarios, the offered itemset is represented in terms of an item (product) table (see Table 4). The itemset is defined extensionally, i.e., all selectable alternatives (items) are enumerated. In our example, five different service configurations (items $i_1..i_5$) can be offered to a user. Table-based representations can be applied if the set of offered items is limited, i.e., the item space is rather small which is often the case, for example, in digital camera or financial service recommendation (Felfernig et al., 2006, 2007). Using a table-based knowledge representation, corresponding database queries can be performed to identify a set of recommendation candidates that support the preferences defined by the user (Felfernig et al., 2006, 2023a). An example of a *user preference* regarding the itemset defined in Table 4 could be *ABtesting* $=$ 1, meaning that the user is interested in a survey software that includes (supports) *ABtesting*. For simplicity, we assume that attributes have a corresponding Boolean domain definition, for example, the domain of attribute *ABtesting* is $\{0, 1\}$ where 1 $=$ *true* (feature included) and 0 $=$ *false*. One exception

TABLE 4 Extensional itemset representation where each item is represented as a table entry assuming the item (table) attributes *ABtesting* (0,1), *statistics* (0,1), *multiplechoice* (0,1), and *license* (0,100).

Item	ABtesting	Statistics	Multiplechoice	License
i_1	0	1	0	0
i_2	0	0	1	100
i_3	1	1	0	100
i_4	0	1	1	100
i_5	1	1	1	100

In our working example, the offered survey software configurations (items $i_1..i_5$) represent the complete set of items (i.e., the product/item catalog). For example, item i_1 represents a survey software configuration (a "free of charge version") which does not support *ABtesting* and *multiplechoice* questions, i.e., only single choice questions can be asked and the *statistics* feature supports a basic analysis of survey data.

TABLE 5 Intensional solution space representation in terms of a set of constraints $\{c_1..c_5\}$ on the CSP variables *ABtesting*, *statistics*, *license*, and *multiplechoice*.

Id	Constraint
c_1	ABtesting=1 $\rightarrow$ statistics=1
c_2	$\neg$(ABtesting=1 $\wedge$ license=0)
c_3	$\neg$(multiplechoice=1 $\wedge$ license=0)
c_4	$\neg$ multiplechoice=1 $\wedge$ $\neg$ ABtesting=1 $\rightarrow$ license=0
c_5	$\neg$(ABtesting=0 $\wedge$ statistics=0 $\wedge$ multiplechoice=0 $\wedge$ license=0)

thereof is the *license* attribute with a domain $\{0, 100\}$ representing the price of a license.

3.1.2 Constraint-based representations

Alternatively, itemsets can be represented in an intensional fashion on the basis of a set of domain-specific constraints (see Table 5), for example, as a *constraint satisfaction problem* (CSP) (Rossi et al., 2006; Felfernig and Burke, 2008). Such knowledge representations are specifically useful if the solution (item) space becomes intractable, i.e., defining and maintaining all alternatives is extremely inefficient and error-prone (or even impossible) and related search queries become inefficient and at least impractical for interactive settings (Falkner et al., 2011).

The constraints in Table 5 represent exactly the solution space, as shown in Table 4. If such constraints are not explicitly known, a given set of items can be defined in terms of one constraint in disjunctive normal form, for example, the entries in Table 4 would be represented as $(ABtesting = 0 \wedge statistics = 1 \wedge license = 0 \wedge multiplechoice = 0) \vee .. \vee (ABtesting = 1 \wedge statistics = 1 \wedge license = 100 \wedge multiplechoice = 1)$—for details, we refer to Felfernig and Burke (2008).

3.1.3 Further knowledge representations

Intensional representations of solution spaces (itemsets) as presented in Table 5 can be implemented with different knowledge representations ranging from *constraint satisfaction problems* (CSPs) (Rossi et al., 2006; Felfernig and Burke, 2008) and *Boolean satisfiability problems* (SAT problems) (Biere et al., 2021; Felfernig et al., 2021) to less frequently used recommendation knowledge representations such as *answer set programming* (ASP) (Eiter et al., 2009; Teppan and Zanker, 2020) and *ontology-based knowledge representations*, for example, *description logics* (DL) (Lee et al., 2006; McGuinness, 2007). In addition, database queries can be applied in such a way that intensionally formulated recommendation knowledge is encoded directly in database queries—see, for example, Felfernig et al. (2006, 2023a). Without loss of generality, in this article, we focus on CSP-based representations when discussing the concepts of constraint-based recommendation. As a basis for our discussion of case-based recommendation approaches, we will use the entries of Table 4.

3.2 Case-based recommendation

3.2.1 Basic approach

Following the basic concepts of case-based reasoning (Kolodner, 2014), case-based recommendation uses a knowledge-rich representation of the item domain and can therefore be classified as a kind of knowledge-based recommendation (Burke, 2000; Khan and Hoffmann, 2003; Lorenzi and Ricci, 2005). In a basic setting, the item assortment can be regarded as the case base, and recommendations are determined by identifying those items from the case base which "support" the user preferences (Lorenzi and Ricci, 2005). In this context, product features (also denoted as properties or attributes) are used to specify user preferences. The more preferences are supported by an item, the higher its user relevance. Examples of case-based recommendation approaches going beyond an equality match between item properties and user preferences are the following—see, for example, Lorenzi and Ricci (2005).

3.2.2 Interest confidence value

This approach is based on the idea of predicting item relevance on the basis of the similarity of a new item with items a user has already evaluated positively in the past (Lorenzi and Ricci, 2005; Musto et al., 2015). This approach is in the line with the ideas of content-based recommendation where the similarity between a new item and properties of already consumed items is used to estimate recommendation relevance. A simple example of the

TABLE 6 Example of an *interest confidence value* (*icv*)-based evaluation of the user relevance of new items (in_k) on the basis of the similarity with already consumed items i_l.

Consumed items	New items				
	in_1	in_2	in_3	in_4	in_5
i_1	0.2	0.4	0.3	0.9	0.5
i_2	0.4	0.6	0.2	0.6	0.7
i_3	0.4	0.1	0.5	0.5	0.6
icv (avg)	0.33	0.36	0.33	0.66	0.6

For example, the similarity between the consumed item i_1 and the new item in_2 is assumed to be 0.4. In this example, item in_4 has the highest *interest confidence value* (0.66) and therefore can be regarded as a user-relevant item.

application of interest confidence values (*icv*) is shown in Table 6, where the *average* (*avg*) *similarity* between a new item (the user did not see up to now) and (consumed) items positively evaluated in the past is used to estimate item relevance.

3.2.3 Attribute-level similarity metrics

This approach is based on the idea that the degree of satisfaction of individual user requirements has a direct impact on the corresponding item ranking (Lorenzi and Ricci, 2005). In such contexts, attribute-level similarity metrics (McSherry, 2003) can be used to determine the similarity between the user requirements and the items included in the product assortment. For example, if explicit price information of an item is available, the *less is better* (LIB) similarity metric can be applied (the lower the price, the higher the item ranking). The *more is better* (MIB) similarity metric can be applied in the context of technical attributes, such as the resolution of a digital camera or the return rate of a financial service (McSherry, 2003; Felfernig et al., 2013c) (the higher the attribute value the better). *Equal is better* (EIB) is used in contexts where users explicitly require a specific attribute value (e.g., the color of a car), and *nearer is better* (NIB) can be used in situations where an item should fulfill a specific requirement as good as possible, for example, the size of a TV screen in the living room. The overall similarity between user requirements and an item is then determined on the basis of a "global" similarity that combines individual attribute-level similarity functions. In our working example, the *license* attribute could be associated with an *LIB*, the remaining attributes with an *EIB* attribute-level similarity.

3.2.4 Refine, relax, and compromise

When querying a case base (i.e., an item catalog) with a set of user preferences, it can happen that the number of candidate items (items that satisfy the user preferences) is (1) too large and—as a consequence—the initial set of user requirements needs to be refined or (2) too small (in the worst case, no solution could be identified), which means that the user requirements have to be relaxed. In the first case, the set of user requirements needs to be extended, for example, if a user has specified the requirement $statistics = 1$, this results in four candidate items ($\{i_1, i_3, i_4, i_5\}$—see Table 4). Refining the original query to $statistics = 1 \wedge ABtesting = 1$ reduces the set of candidate items to $\{i_3, i_5\}$. In contrary, if a user has specified the requirement $ABtesting = 1 \wedge license = 0$, the corresponding query would result in an empty set. The user has two options to resolve the inconsistency, i.e., to relax the query: either to exclude *ABtesting* or to accept a *license* payment, i.e., $license = 100$. Such trade-off decisions can be supported by so-called *compromise-based user interfaces* which group items with regard to different possible compromises (McSherry, 2003). Such interfaces would group and explain candidate items in the line of, for example, *most of your requirements are fulfilled, however, these items require a license payment*.

3.2.5 Critiquing

Critiquing-based recommendation originates from different case-based recommenders (Kolodner, 1992; Bridge et al., 2005) which often support a search-based approach where—depending on a set of defined user preferences—the system recommends items which support in one way or another those preferences. While basic case-based and constraint-based recommendation focus on supporting a *search-based recommendation process*, critiquing-based recommender systems (as a specific type of case-based recommender system) follow a *navigation-based approach* (Chen and Pu, 2012), which focuses on better supporting users in *exploring* the solution (item) space. Critiquing-based recommendation is based on the idea of presenting *example (reference) items* to a user who then can (1) accept the proposed item or (2) define critiques in terms of changes that are needed to make an item acceptable. Existing critiquing-based recommenders are based on a predefined itemset (product table)—see Table 4. These systems are regarded as knowledge-based, since items are associated with semantic properties, and user-defined (selected) critiques can be regarded as logical criteria (constraints) to be fulfilled by recommendations.

3.2.6 Unit critiquing

Let us assume that in the context of our working example a critiquing-based recommender supports six basic critiques which are (1) *furtherstat*, (2) *reducestat*, (3) *excludemc*, (4) *includemc*, (5) *excludelicense*, and (6) *includelicense*. The critiquing-based approach used in this example is *unit-critiquing* (Chen and Pu, 2012; Mandl and Felfernig, 2012), where in each critiquing cycle, a critique on an individual item attribute is specified. Table 7 shows how critiques can be translated into corresponding item selection criteria. For example, if a user selects the critique *further statistics*

TABLE 7 Translating critiques into item selection criteria where *mc=multiplechoice*, *stat=statistics*, and "$\mapsto$" represents a mapping operator from critiques to item selection criteria.

Critique	Item selection criteria
Furtherstat	$Statistics = 0 \mapsto statistics = 1$
	$Statistics = 1 \mapsto ABtesting = 1$
Reducestat	$ABtesting = 1 \mapsto ABtesting = 0$
	$Statistics = 1 \wedge ABtesting = 0 \mapsto statistics = 0$
Excludemc	$Multiplechoice = 1 \mapsto multiplechoice = 0$
Includemc	$Multiplechoice = 0 \mapsto multiplechoice = 1$
Excludelicense	$License = 100 \mapsto license = 0$
Includelicense	$License = 0 \mapsto license = 100$

For example, if the critique *excludelicense* is specified by the user, an item supporting the selection criteria $license = 0$ will be shown.

(*furtherstat*) and statistics is not included in the shown reference item, the corresponding selection criteria would be $statistics = 1$. Furthermore, if statistics is already included in the reference item, $ABtesting = 1$ would be defined as additional selection criteria (see the entries in Table 7).

Table 8 shows an example of a *unit critiquing session* where the initial (reference) item presented to the user is i_1. The user specifies the critique *furtherstat* which requires the inclusion of additional statistic features into the survey software service configuration. Taking into account, this critique means to combine the properties of the reference item i_1 ($ABtesting = 0, statistics = 1, license = 0, multiplechoice = 0$) with the selection criteria derived from Table 7 (i.e., $ABtesting = 1$), resulting in a new set of criteria ($\{ABtesting = 1, statistics = 1, license = 0, multiplechoice = 0\}$). There does not exist an item in $\{i_1..i_5\}$ which completely fulfills these criteria. A related tradeoff is to choose an item as a new reference item which is mostly similar to the current selection criteria—in our case, this is item i_3 which only requires the adaptation of $license = 0$ to $license = 100$. The user again specifies a critique (*includemc*) which results in a new set of criteria ($\{ABtesting = 1, statistics = 1, license = 100, multiplechoice = 1\}$), leading to the new reference item i_5 which is finally accepted by the user.

The idea of critiquing-based recommendation is to identify an item which is similar to the current item but (in addition) takes into account the criteria specified by the critique. In our example, we have assumed that the search for a new item returns an exact match (*EIB* metric), i.e., all search criteria are fulfilled. However, the search for new reference items is often designed more flexible, for example, other attribute-level similarity metrics can be applied to determine new reference items (Lorenzi and Ricci, 2005).

3.2.7 Further critiquing approaches

For demonstration purposes, we have discussed the concepts of unit critiquing in more detail. However, critiquing-based recommender systems support various alternative types of critiques (Chen and Pu, 2012; Güell et al., 2020). The idea of *compound critiques* (Smyth et al., 2004; Zhang et al., 2008) is to allow the definition of critiques which refer to more than one item property. A related example would be the compound critique *furtherstat and includelicense*, indicating the interest of a user in further statistics features also accepting (additional) license costs. A major advantage of compound critiquing is that due to the specification of multiple criteria in one critiquing cycle, significantly larger "jumps" in the itemspace are possible (Aggarwal, 2016) and—as a result—less critiquing cycles are needed to find a solution. As an extension of compound critiques, *dynamic critiques* (Reilly et al., 2004; McCarthy et al., 2005) are based on the idea of mining frequent critiquing patterns which are then presented to a user.

3.2.8 Handling inconsistent search criteria

Critiquing-based recommender systems try to take into account the critiques defined by a user. Since user preferences (search criteria) typically change within the scope of a recommendation session, inconsistencies can occur (Atas et al., 2021). For example, a user is first interested in a survey software configuration without license fee resulting in the reference item i_1. Then, the user wants to additionally include the *ABtesting* feature which then results in an inconsistency, since no item supports both, no license fee and *ABtesting* at the same time. From the logical point of view, such (inconsistent) search criteria are constraint sets $CRIT = \{crit_1..crit_q\}$ and in the case of inconsistencies in $CRIT$, one option is to determine explanations (diagnoses) which indicate different options of resolving an inconsistency.[6] Alternatively, weighting schemes are applied in such contexts meaning that "elder" critiques have a higher probability of being deleted from a critiquing history. Another alternative is the retrieval of a new reference item which is as much as possible similar to the current set of search criteria (see our example in Table 8).

3.2.9 Examples of case-based recommender systems

Table 9 provides an overview of example case-based recommender applications. (Burke et al., 1996) introduce the FINDME approach to support a unit critiquing-based navigation in complex information spaces—discussed item domains are cars, videos, and apartments. The ENTREE system (Burke, 2000) supports unit critiquing-based recommendations in the restaurant domain, and this system is also based on the FINDME approach introduced by (Burke et al., 1996). DIETORECS (Fesenmaier et al., 2003) is a case-based recommender system that supports a similarity-based approach to case retrieval, i.e., users have to specify their requirements and select a case of relevance which is then used as a basis for further search. RECOMMENT (Grasch et al., 2013) is a natural language-based user interface with an underlying unit-critiquing-based recommender system. (Musto et al., 2015) introduce an asset allocation strategy framework which supports the recommendation of asset classes (e.g., Euro

6 For details, see the following discussion of constraint-based recommender systems.

TABLE 8 A simple example of a unit-critiquing session consisting of two critiquing cycles.

Attribute	Ref. item (i_1)	Ref. item after cycle 1 (i_3)	Ref. item after cycle 2 (i_5)
ABtesting	0	1	1
Statistics	1	1	1
License	0	100	100
Multiplechoice	0	0	1
Unit critique	(1) further statistics	(2) includemc	Accept

In critiquing cycle (1), the critique *further statistics* is specified on the reference item i_1, in cycle (2), *includemc* is specified as a critique. After cycle (2), the user is satisfied with (accepts) the recommended item (i_5).

TABLE 9 Overview of example case-based recommender applications.

Recommender application	Recommendation domain	References
RENTME	Apartments	Burke et al., 1996
ENTREE	Restaurants	Burke, 2000
DIETORECS	Travel plans	Fesenmaier et al., 2003
RECOMMENT	Digital cameras	Grasch et al., 2013
Asset allocation framework	Financial services	Musto et al., 2015
EHEARSS	Healthcare	Lee and Kim, 2015
Recommender for Recreational Runners	Training plans	Feely et al., 2020
CEBRA	Financial services	Hernandez-Nieves et al., 2021
Smart City Initiative	Smart cities	Bokolo, 2021

Bond, High Yield Bond, Euro Stocks, and Emerging Market Stocks) that fulfill the goals of the investor and are basically generated on the basis of an analysis of the portfolios of similar clients (investors). Lee and Kim (2015) introduce EHEARSS which is a case-based recommender system in the healthcare domain which helps to identify, for example, relevant doctors, depending on the current (aggregated) health record and those of similar cases. In this context, an ontology helps to assure the correctness of recommendations, for example, in terms of the proposed medical diagnosis. Feely et al. (2020) introduce a case-based recommendation approach for the recommendation of marathon training plans and race pacings based on case information about the workouts and race histories of similar runners. CEBRA (Hernandez-Nieves et al., 2021) supports the recommendation of banking products using a basic case-based reasoning recommender operating on a user's financial service profile and corresponding demographic data. Finally, the smart city initiative (Bokolo, 2021) focuses on the case-based recommendation of smart city dimensions, i.e., by taking a look at similar cities, corresponding smart city-related activities for the current city are determined in order to achieve predefined sustainability goals, for example, on the basis of the best use of technological and human resources.

3.3 Constraint-based recommendation

The concept of constraint-based recommendation (Felfernig and Burke, 2008) is based on the idea that recommendation knowledge is represented in terms of a set of variables and a corresponding set of constraints. As already mentioned, such constraints can be defined in an extensional fashion (by just enumerating the items part of the solution space) or in an intensional fashion where the recommendation space is represented in terms of a set of logical formulae. In this context, a constraint-based recommendation task (CB-REC TASK) can be defined as follows (see Definition 1).[7]

Definition 1. A constraint-based recommendation task (CB-REC TASK) can be defined as a constraint satisfaction problem (CSP) (V, C, R) where $V = \{v_1..v_n\}$ is a set of finite domain variables with associated variable domain definitions $dom(v_i)$, $C = \{c_1..c_m\}$ is a set of constraints, and $R = \{r_1..r_k\}$ is a set of user requirements.

Following Definition 1, an example recommendation task (V, C, R) based on the entries in Table 5 is as follows: $V = \{ABtesting, statistics, multiplechoice, license\}$, $dom(ABtesting) = dom(statistics) = dom(multiplechoice) = \{0, 1\}, dom(license) = \{0, 100\}$, $C = \{c_1..c_5\}$, and $R = \{r_1 : ABtesting = 1\}$, assuming that the user is interested in the *ABtesting* feature.

A constraint-based recommendation CB-REC for a given constraint-based recommendation task (CB-REC TASK) can be defined as follows (see Definition 2).

Definition 2. A constraint-based recommendation (CB-REC) for a defined CB-REC TASK is a set of tuples $REC = \bigcup_{i_\alpha \in I}\{(rank_{i\alpha}, i_\alpha)\}$ where $rank_{i\alpha}$ represents the recommendation rank assigned to item $i_\alpha \in I$ defined by a recommendation function rf and $\forall(rank_{i\alpha}, i_\alpha) \in REC$: consistent$(C \cup R \cup a(i_\alpha))$ where $a(i_\alpha)$ denotes the variable value assignments associated with item i_α.

Assuming $R = \{r_1 : ABtesting = 1\}$, a CB-REC in our working example could be $REC = \{(1, i_5), (2, i_3)\}$ where $a(i_3) = \{ABtesting = 1, statistics = 1, license = 100, multiplechoice = 0\}$ and $a(i_5) = \{ABtesting = 1, statistics = 1, license = 100, multiplechoice = 1\}$.

7 For simplicity, we omit a differentiation between variables describing user preferences and those describing item properties and different constraints between these variables—for related details, we refer to Felfernig et al. (2006) and Felfernig and Burke (2008).

3.3.1 Ranking items

Up to now, we did not specify a function rf for the ranking of the items in REC. A simple ranking function could just count the number of supported features (see Equation 1), which would result in the mentioned recommendation ranking of $REC = \{(1, i_5), (2, i_3)\}$ since the number of supported features of i_5 is 3 [$support(i_5) = 3$] whereas $support(i_3) = 2$. In other words, item i_3 is outperformed by one item resulting in a ranking of 2.[8]

$$rf(i_\alpha) = 1 + |\{i_k \in REC(k \neq \alpha) : support(i_k) > support(i_\alpha)\}| \quad (1)$$

An alternative to our simplified ranking function (Equation 1) is to introduce a utility function which evaluates utility of individual items on the basis of a pre-defined set of interest dimensions (Felfernig et al., 2006, 2018a). In our software service recommendation scenario, examples of relevant interest dimensions are *economy* and *quality* (see also Table 10).

Such utility-based evaluation schemes can then be used by a utility function to determine the overall utility of individual items—see, for example, Equation (2)— where D represents a set of interest dimensions [in our case, $D = \{economy, quality\}$] and $eval(i_\alpha, d)$ represents the evaluation scheme as presented in Table 10. Assuming equal importance of the two example interest dimensions [e.g., $importance(economy) = 0.5$ and $importance(quality) = 0.5$], $utility(i_3) = 0.0 + 10.0 = 10.0$ and $utility(i_5) = 0.0 + 15 = 15.0$, i.e., item i_5 has a higher utility compared with item i_3. Depending on the user-specific importance of individual interest dimensions, the resulting utility values can differ.

$$utility(i_\alpha) = \Sigma_{d \in D}\, eval(i_\alpha, d) \times importance(d) \quad (2)$$

3.3.2 Dealing with inconsistent requirements

In constraint-based recommendation, it can be the case that individual user requirements do not allow the determination of a recommendation.[9] For example, if a user is interested in the *ABtesting* and *multiplechoice* features but does not want to pay a license, i.e., $R = \{ABtesting = 1, multiplechoice = 1, license = 0\}$, no solution/item can support this set of requirements. In this example, we are able to identify two different conflicts (Junker, 2004) (see Definition 3), which are minimal sets of requirements that induce an inconsistency.

Definition 3. A conflict (set) $CS \subseteq R$ is a set of constraints with inconsistent($CS \cup C$), i.e., no solution can be found for $CS \cup C$. A conflict set CS is minimal if $\neg\exists CS' : CS' \subset CS$ (subset minimality).

Conflict set minimality is important due to the fact that just one requirement needs to be deleted (i.e., relaxed) from CS in order to resolve the conflict. In our example, there exist two minimal conflict sets which are CS_1:$\{ABtesting = 1, license = 0\}$ and CS_2:$\{multiplechoice = 1, license = 0\}$. To resolve the conflict CS_1, a user can choose between the two options of excluding *ABtesting* ($ABtesting = 0$) or paying a *license* ($license = 100$). The same approach can be followed to resolve CS_2, i.e., to either accept $multiplechoice = 0$ or accept $license = 100$.

Such conflict sets can be used in interactive recommendation settings to support users in figuring out ways from the *no solution could be found* dilemma. A set Δ of requirements is denoted as diagnosis (Reiter, 1987; Felfernig et al., 2009b, 2012) if it helps to resolve all identified conflicts (see Definition 4).

Definition 4. A diagnosis $\Delta \subseteq R$ is a set of constraints with consistent($R - \Delta \cup C$), i.e., at least one solution can be found for $R - \Delta \cup C$. A diagnosis Δ is minimal if $\neg\exists\Delta' : \Delta' \subset \Delta$ (subset minimality).

Both concepts, i.e., *conflict sets* and corresponding *diagnoses* can be used to support users in inconsistent situations. Conflict sets are helpful in the context of *repeated conflict resolution* (for example, when purchasing a new car for the family, the preferences of individual family members change over time), and diagnoses can be used when users are interested in *quick repairs* (for example, when parametrizing an operating system, some inconsistent parameter settings need to be adapted).

3.3.3 Examples of constraint-based recommender systems

Table 11 provides an overview of example constraint-based recommender applications. FSADVISOR (Felfernig and Kiener, 2005) is a financial service recommender system that proposes new financial services depending on the current financial status and requirements of the customer. In the line of FSADVISOR, VITA (Felfernig et al., 2007) is a financial service recommender system supporting sales representatives in the preparation and conduction of customer dialogs. Both systems are based on a database query-based approach implemented in ADVISORSUITE, which is a development environment for constraint-based recommender applications (Felfernig et al., 2006; Jannach and Kreutler, 2007). In addition, MORTIMER is a constraint-based recommender application developed on the basis of ADVISORSUITE which focuses on the constraint-based recommendation of cigars. AUTHENTIC (Murphy et al., 2015) is a constraint satisfaction-based recommendation environment for supporting the achievement of energy saving goals on the basis of adapted schedules for activating household appliances. Recommendations of behavior change (e.g., postponed activation of a dishwasher to exploit more self-produced energy) are determined on the basis of energy consumption-related sensor data. Wobcke et al. (2015) introduce a P2P recommendation environment in the context of online dating scenarios where constraints (rules) are used to define baseline recommendation strategies. RECTURK (Ulz et al., 2017) is a framework for the development of constraint-based recommender applications. The underlying idea is to apply the concepts of *human computation* (Law and von Ahn, 2011) to alleviate the definition and maintenance of recommendation rules. Almalis et al. (2018) introduce a constraint-based job recommender system where constraints are used to assure defined compatibilities between job seekers and job announcements. Finally, Esheiba et al. (2021)

8 In this context, *license* is not regarded as a feature.

9 In critiquing-based recommendation, such situations occur if search criteria become inconsistent.

TABLE 10 A simple utility-based evaluation scheme for recommendation candidates, for example, paying no license fee, i.e., *license* = 0, contributes to the evaluation dimension economy.

Interest dimension	ABtesting		Statistics		License		Multiplechoice	
	0	1	0	1	0	100	0	1
Economy	0	0	0	0	10	0	0	0
Quality	0	10	0	10	0	0	0	10

TABLE 11 Overview of example constraint-based recommender applications.

Recommender application	Recommendation domain	References
FSADVISOR	Financial services	Felfernig and Kiener, 2005
ADVISOR SUITE	Digital cameras	Felfernig et al., 2006
MORTIMER	Cigars	Zanker et al., 2006
VITA	Financial services	Felfernig et al., 2007
AUTHENTIC	Energy saving	Murphy et al., 2015
P2P framework	Online dating	Wobcke et al., 2015
RECTURK framework	Digital cameras	Ulz et al., 2017
FODRA framework	Jobs	Almalis et al., 2018
PSS RECOMMENDER	Product services	Esheiba et al., 2021

present a constraint-based recommender system supporting the filtering-out of irrelevant product services and the ranking of the remaining relevant ones.

4 Recent advances in knowledge-based recommendation

In this section, we provide an overview of the recent advances in knowledge-based recommendation (compared with the basic approaches presented in Section 3). We discuss the following aspects: (1) Users are often in the situation of not knowing every detail about a product and—as a consequence—are not able to make a related informed decision. For example, if someone is interested in digital cameras, he/she might be able to specify preferences regarding attributes such as *resolution*, *maximum price*, *weight*, and *primary application* (e.g., landscape photography or sports) but at the same time might not be able to specify preferences regarding attributes such as *maximum frames per second* or *maximum exposure time*. In our example, users might not be aware of the usefulness of the *ABtesting* feature. Related techniques to support users in such contexts are presented in Section 4.1. (2) Given a set of user preferences, situations can occur where no solution can be identified (see also Section 3)—in Section 4.2, we introduce concepts for personalized conflict resolution. (3) In some situations, users have adaptation (reconfiguration) requirements regarding already purchased items. For example, after having purchased an online survey software license, a user detected the relevance of the *ABtesting* feature and is interested in a feature extension. Related technical solutions are presented in Section 4.3. (4) Knowledge-based recommendation is not only relevant in recommendation scenarios focusing on single users but also in scenarios where recommendations have to be determined for groups, for example, planning round-trips for tourist groups or deciding about the features to be included in a software or service. Aspects of knowledge-based group recommender systems are presented in Section 4.4. (5) In Section 4.5, we discuss further approaches focusing on the topics of hybrid recommendation, search optimization, explanations, recommendation knowledge acquisition, conversational recommendation, and explanations.

4.1 Recommending preference settings

There are situations where users need help in specifying preferences regarding specific item properties. Table 12 depicts a simplified example of an attribute value proposal in the context of a *case-based recommendation* scenario: the *current* user has already specified his/her preferences regarding the attributes *multiplechoice* and *statistics*. Using a nearest neighbor (NN)-based approach, i.e., determining the session with the most similar preferences, results in session s_3. Consequently, we can recommend the attribute values $ABtesting = 1$ and $license = 100$ to the *current* user. The same approach to attribute value recommendation can also be applied (1) in the context of customer segmentation (recommendation) where customers have to be identified who will be interested in a new item feature (Felfernig et al., 2021) and (2) to figure out features (attributes), a customer is interested in specifying—this is relevant in scenarios where users are not interested in specifying all attributes (Mirzadeh et al., 2005; Dragone et al., 2018).

An alternative to NN-based recommendation is to apply model-based approaches such as matrix factorization or neural networks—for related details see Erdeniz et al. (2019, 2022) and Uta et al. (2022).

Beyond case-based recommendation, such approaches to attribute value recommendation are also applied in *constraint-based recommendation* (and beyond) (Felfernig and Burke, 2008; Falkner et al., 2011; Fargier et al., 2016; Pereira et al., 2016; Temple et al., 2017). A major issue in such contexts is the fact that a recommendation of one or more attribute settings could be *inconsistent* with the underlying set of constraints (C). One way to assure consistency is to pre-determine a set of nearest neighbors and to show recommendations to users after having tested recommendation consistency (Cöster et al., 2002; Felfernig and Burke, 2008; Falkner et al., 2011; Pereira et al., 2016).

TABLE 12 Example user sessions s_i representing "successfully" completed recommendation sessions (e.g., a user has selected a specific item).

Session	ABtesting	Statistics	Multiplechoice	License
s_1	1	1	0	100
s_2	0	1	0	0
s_3 (NN)	1	1	1	100
Current	?	1	1	?

In this example, user (session) s_3 can be regarded as the nearest neighbor (NN) of the *current* user, i.e., both, *ABtesting* and *license* are recommended to be included.

An alternative is to learn *solver search heuristics* in such a way that constraint solvers are enabled to predict user-relevant attribute value settings (Uta et al., 2022), i.e., to directly integrate recommendation knowledge into constraint solver search, for example, by applying neural networks that receive as input a set of user preferences and propose corresponding solver search heuristics [in terms of variable (value) orderings] as output. In the example shown in Table 12, variable value orderings for the not yet instantiated variables *ABtesting* and *license* would be *ABtesting* [1,0] and *license* [100,0], indicating that the solver should first try to instantiate these variables with 1 (100) before trying other instantiations. For an overview on the integration of machine learning and constraint solving, we refer to Popescu et al. (2022).

In *critiquing-based recommendation*, such attribute value recommendations could be also used in the initial preference specification phase. However, critiquing-based recommendation follows the idea of critiquing a shown reference item. Specifically, in critiquing scenarios, attribute value recommendation is "replaced" by the recommendation of critiques (McCarthy et al., 2010; Mandl and Felfernig, 2012; Murti et al., 2016; Xie et al., 2018)—see the simplified unit critique recommendation example depicted in Table 13. In this simplified example,[10] the critiquing session most similar to the critiquing session of the *current* user (session) is cs_2. This can be determined, for example, by determining the intersection of the critiquing sets of cs_i and *current* where $|critiques(cs_1) \cap critiques(current)| = 1$ and $|critiques(cs_2) \cap critiques(current)| = 2$. Using a nearest neighbor-based approach, critiques and item selection of cs_2 can be used to recommend future critiques to the *current user* but also to predict items that will be chosen by the current user (Mandl and Felfernig, 2012).

4.2 Dealing with "no solution could be found" situations

As discussed in Section 3, users of knowledge-based recommenders in some situations need support to get out of the *no solution could be found dilemma*. Approaches that can proactively support users in such contexts are conflict detection (Junker, 2004) and model-based diagnosis (Reiter, 1987; Felfernig et al., 2012; Walter et al., 2017). Such algorithms help users to understand trade-offs in the current situation and propose different options to resolve inconsistencies. Having user preference weights available, for example, in terms of explicitly defined preference weights, algorithms can be applied in different ways to determine preferred (personalized) diagnoses. For demonstration purposes, we assume the following preference ordering (see Table 14) for our example conflict sets $CS1: \{ABtesting = 1, license = 0\}$ and $CS2$: $\{multiplechoice = 1, license = 0\}$ derived from the user requirements $R = \{ABtesting = 1, multiplechoice = 1, license = 0\}$.

Following a simple additive utility-based scheme (Felfernig et al., 2013c) using the orderings depicted in Table 14 (the lower the ordering position, the higher the importance of the related user requirement), the conflicts CS_1 and CS_2 would be resolved (in a personalized fashion) as follows: for the user in Session s_1 (user 1), we would keep the exclusion of *license* as-is and exclude both, *ABtesting* and *multiplechoice* ($1 < 3 + 4$). Vice-versa, in the case of session s_2 (user 2), we would keep the preferences regarding *ABtesting* and *multiplechoice* as-is and accept the inclusion of a license fee ($1 + 2 < 4$). In a similar fashion, such an approach can be applied in critiquing-based recommendation sessions where—on the basis of importance information about individual critiques [user sentiments about individual attributes could be available or "older" critiques could be interpreted as less important (Chen et al., 2017)]—corresponding conflict resolutions could be proposed.

4.3 Recommendations for reconfiguration

In scenarios where users have already purchased an item and want to extend the item afterward, recommendations for reconfigurations are required, i.e., which attributes of the original solution need to be adapted such that the new user requirements can be taken into account (Felfernig et al., 2018b; Weckesser et al., 2018). For example, if a user has already ordered a specific survey software service consisting of the feature $\{statistics = 1\}$ (item i_1) but then detects that he/she wants to include also the *ABtesting* feature, the question arises which of the initial attribute settings (user preferences) remain the same and which ones need to be adapted. In such scenarios, critiquing-based recommendation would not be the primary choice since reconfiguration is in the need of *minimal changes* to an already ordered item. In contrast, case-based reasoning can be applied by using cases (items) with an exact attribute-wise match to the already purchased item with a minimal number of needed adaptations.

If we assume the availability of a product table as shown in Table 4 and a user who has purchased item i_1 and now wants to extend this configuration with *ABtesting*, this results in the search criteria $\{ABtesting = 1, statistics = 1, multiplechoice = 0, license =$

10 For simplicity, all critiquing sessions have the same number of critiques, however, in real-world scenarios, the number of critiques can very significantly between different critiquing sessions.

TABLE 13 Example critiquing sessions cs_i representing "successfully" completed recommendation sessions.

Session	Reference item	$crit_1$	$crit_2$	$crit_3$	Selection
cs_1	i_1	$i_1.furtherstat$	$i_4.reducestat$	-	i_1
cs_2 (NN)	i_1	$i_1.includemc$	$i_4.furtherstat$	$i_5.excludemc$	i_3
Current	i_1	$i_1.includemc$	$i_7.furtherstat$	?	?

In this example, session cs_2 can be regarded as the nearest neighbor (NN) of the current user. The reference item i_1 is presented in Table 4.

TABLE 14 Example preference orderings for requirements (the lower the more important the preference).

Session	ABtesting	Statistics	Multiplechoice	License	Diagnosis Δ
s_1	3	2	4	1	$\{ABtesting = 1, multiplechoice = 1\}$
s_2	2	3	1	4	$\{license = 0\}$

For example, in session s_1, keeping the selected *license* type as-is has the highest priority.

TABLE 15 Simplified reconfiguration setting: a user has already purchased item i_1 and—after some time—wants to extend this configuration by including *ABtesting*.

Item	ABtesting	Statistics	Multiplechoice	License
i_1 (purchase)	0 (1)	1	0	0
i_2	0	0	1	100
i_3 (NN)	1	1	0	100
i_4	0	1	1	100
i_5	1	1	1	100

0}. The corresponding nearest neighbor configuration is item i_3 [based on the *equal is better* (EIB) similarity metric]—see Table 15. In addition to the inclusion of *ABtesting*, the selection of item i_3 requires one additional adaptation which is $license = 100$. Beyond counting the number of adaptations, reconfiguration can also take into account sentiments (Chen et al., 2017): if additional features could be included without further costs or efforts, this should be preferred over reconfigurations where those features are not included.

In constraint-based recommendation, $i = \{a_1..a_k\}$ can be regarded as a set of variable assignments (constraints) describing the original item i (in our example, i_1); furthermore, $R = \{r_1..r_l\}$ is a set of reconfiguration requirements (in our example, $R = \{ABtesting = 1\}$). A diagnosis $\Delta \subseteq i$ is a set of variable assignments in i that need to be adapted such that a solution for the reconfiguration task can be identified (Felfernig et al., 2018b). More formally, $\Delta \subseteq i$ represents a diagnosis if consistent $(i - \Delta \cup C \cup R)$, i.e., the corresponding domain-specific constraints C (see Table 5) need to be taken into account (Felfernig et al., 2018b). If weights for user preferences are available (see, e.g., Table 14), reconfiguration can be personalized, i.e., for individual users having purchased the same original item, different reconfigurations could be proposed. Following the standard approach of conflict detection (Junker, 2004) and diagnosis identification (Reiter, 1987), a reconfiguration can be determined the same way as shown in the context of identifying minimal sets of inconsistent requirements. If a user has originally selected (purchased) item i_1, a corresponding reconfiguration requirement $R = \{ABtesting = 1\}$ would induce the (singleton) conflicts CS_1: $\{ABtesting = 0\}$ and CS_2: $\{license = 0\}$, resulting in a corresponding reconfiguration $\{ABtesting = 1, statistics = 1, multiplechoice = 0, license = 100\}$ (Felfernig et al., 2018b).

4.4 Knowledge-based recommendation for groups

Group recommender systems (Pessemier et al., 2017; Felfernig et al., 2024b) are typically built on the top of single user recommender systems following one of the approaches of (1) *aggregated predictions* or (2) *aggregated models*. With aggregated predictions, the items recommended to individual group members can be merged. In the context of aggregated models, the preferences of individual group members are merged, resulting in a group preference model which is then used to determine recommendations.

In *case-based recommendation* for groups, the determination of group recommendations can be based on two different approaches. With *aggregated predictions*, in a first step, relevant recommendations would be determined for each user (see Table 16). In the following, these individual recommendations (items) have to be aggregated, according to a specific aggregation strategy. For example, if we apply *majority voting*, the overall recommendation for the group would be item i_5 since it has received more recommendations compared with other recommendation candidates.

If the recommender system is based on *aggregated models* (see Table 17), user-individual preferences have to be aggregated into a group model which can then be used for determining

TABLE 16 Example of user-individual item recommendations as a basis for determining a group recommendation using the *aggregated prediction* approach (with majority voting, i_5 gets recommended).

User	Item
u_1	i_5
u_2	i_4
u_3	i_1
u_4	i_5

recommendations. In the example of Table 17, the overall case-based group recommendation would be item i_3 since this one exactly represents the preferences of the determined group model.

We want to emphasize that following the approach of *aggregated models* can result in situations where no exact match between the attribute settings in the group model and corresponding item properties is possible. However, in such cases, the applied similarity functions will recommend items most similar to the preferences stored in the group model.

If a *critiquing-based recommendation* approach (McCarthy et al., 2006) is used, an *aggregated prediction* approach can be implemented in such a way that each group member completes an individual critiquing process, and the resulting items are then merged by an aggregation function. Following an *aggregated models* approach, individual critiques can be merged, for example, after one or more critiquing cycles, in order to show a new reference item (to individual group members) which is regarded as current recommendation for the whole group—see also McCarthy et al. (2006) and Felfernig et al. (2024b).

In *constraint-based recommendation*, in both cases, i.e., aggregated preferences and aggregated models, a group recommender would have to aggregate the preferences (attribute settings) of the individual group members (Felfernig et al., 2024b). Again, different aggregation approaches on the attribute level are possible—if we follow the goal of achieving fairness (Le et al., 2022), for example, in terms of a nearly equal number of accepted compromises per group member, a corresponding optimization (minimization) problem needs to be solved (see Equation 3). In this context, $p(u_\alpha)$ denotes the preferences of group member u_α, rec denotes the determined group recommendation, and $diff(p(u_\alpha, rec))$ denotes the number of preference tradeoffs to be accepted by group member u_α. This is a group recommendation that takes into account the aspect of *fairness* minimizing tradeoff distances among individual group members.

$$MIN \leftarrow \Sigma_{\{u_i,u_j\}\subseteq Users} |diff\ (p(u_i, rec) - diff\ (p(u_j, rec))| \quad (3)$$

4.5 Further aspects

4.5.1 Hybrid recommendation approaches

The knowledge-based recommendation approaches discussed up to now have—as a central element—an item assortment (i.e., product catalog) or a knowledge base that describes the item assortment in an intensional fashion. Knowledge-based recommendation can also be applied in hybrid recommendation scenarios taking over the role of a supportive/complementary component (Burke, 2002). In many cases, knowledge-based recommender systems take over the role of being responsible of checking domain-specific properties, for example, when recommending medical services, it must be assured that the medical item is compatible with the current state of health of a patient. Furthermore, knowledge bases (e.g., in terms of ontologies and knowledge graphs) can be applied, for example, as a basis for similarity metrics that are able to take into account knowledge structures (Bahramian and Ali Abbaspour, 2015; Colombo-Mendoza et al., 2015; Wang et al., 2019a,b; Sun et al., 2020; Zhu et al., 2020; Sha et al., 2021).

4.5.2 Search optimization

Performance optimization can become an issue specifically in the context of constraint-based recommendation and related conflict detection and diagnosis problem solving (Reiter, 1987; Junker, 2004; Felfernig et al., 2012; Le et al., 2023). Improving the efficiency of constraint solving primarily depends on the ability to select optimal heuristics depending on the given search problem. Optimizing search efficiency is often based on the application of machine learning. A detailed overview of integration scenarios for constraint solving and machine learning is given in the study by Popescu et al. (2022). Specifically in the context of constraint-based recommendation, multi-criteria optimization becomes an issue since (1) constraint-based recommenders are typically applied in interactive scenarios with the need of efficient solution search and (2) at the same time, solutions must be personalized, i.e., take into account the preferences of a user. Thus, constraint-based recommendation is an important application field with the need of an in-depth integration of knowledge-based reasoning and machine learning (Uta et al., 2022).

4.5.3 Knowledge acquisition and maintenance

This is an important aspect specifically in the context of constraint-based recommendation where recommendation knowledge is encoded in terms of constraints, rules, or database queries (Jannach and Kreutler, 2007; Felfernig et al., 2009a, 2013b). It must be ensured that the defined constraints correspond with real-world constraints, i.e., reflect the item domain and recommendation knowledge. The related phenomenon of the *knowledge acquisition bottleneck*, i.e., significant communication overheads between domain experts and knowledge engineers, is still omnipresent when applying such knowledge-based systems. Related research efforts focus on the topics of *automated testing and debugging*, *human-centered knowledge acquisition*, and *deep models of knowledge understanding*. Automated testing and debugging focuses on the application of model-based diagnosis where pre-defined test cases are used to induce conflicts in the knowledge base which are then resolved on the basis of diagnosis algorithms—see, for example, Le et al. (2021). Understanding the complexity individual knowledge structures can also help to increase the maintainability of knowledge bases—this topic includes the aspects of cognitive complexities of knowledge structures (Felfernig et al., 2015) and related knowledge structuring, for example, in terms

TABLE 17 Example of a group recommendation based on the *aggregated model* approach resulting in the recommendation of item i_3.

Session	ABtesting	Statistics	Multiplechoice	License
u_1	1	1	0	100
u_2	0	1	0	0
u_3	1	1	1	100
Group model	1	1	0	100

of the ordering of constraints in a recommendation knowledge base (Felfernig et al., 2013a). Finally, human-computation-based knowledge acquisition (Ulz et al., 2017) and user query recommendation (Daoudi et al., 2016) make knowledge definition accessible to domain experts by asking simple questions, for example, *given the user requirements of statistics* $= 1$ *and license* $=$ 100*, which items can be recommended?* From the answers to such questions, a recommendation knowledge base can be generated (Ulz et al., 2017).

4.5.4 Conversational recommendation

Following the characterizations of Christakopoulou et al. (2016), Wu et al. (2019), Cordero et al. (2020), Dong et al. (2020), Zhou et al. (2020), and Jannach et al. (2021) conversational recommender systems are interactive systems supporting users in the navigation of the item space in an efficient fashion and recommend items based on a systematic approach of preference elicitation. Knowledge-based recommenders can be regarded as conversational since users are guided through a preference elicitation dialog (Zou et al., 2020). However, further types of conversational recommender systems exist which extend existing approaches specifically with more flexible types of preference elicitation using, for example, natural language dialogs (Grasch et al., 2013), chatbot technologies (Tazl et al., 2019), and specific interfaces based on large language models (Dai et al., 2023).

4.5.5 Explanations in knowledge-based recommendation

In the context of recommender systems, explanations can have various goals such as persuading users to purchase an item, to increase the level of trust in a recommendation, or simply to extend a user's item domain knowledge (Tintarev and Masthoff, 2012). In this context, basic explanation types are the following: (1) *why* explanations help a user to understand the reasons why a specific item has been recommended. In the context of single-shot recommendation approaches such as collaborative filtering and content-based filtering, explanations are directly related to the used algorithmic approach. For example, *this item is recommended, since similar users also liked it* or *this item is recommended, since you liked similar items in the past*. Answering such *why* questions in the context of knowledge-based recommendation means to relate recommendations to user preferences, for example, *this camera is recommended since it includes a high frame rate per second which corresponds to your requirement to be able to perform sports photography on a professional level*. This example also shows that recommendation dialogs do not necessarily include technical item properties but could also support scenarios where users specify *high-level preferences* (Felfernig et al., 2006) which are then translated into technical properties by the recommender system. (2) *Why not* explanations help users to understand in more detail why no solution could be found for their requirements. In this context, conflicts (Junker, 2004) help to understand, for example, individual incompatibilities between user requirements, whereas diagnoses (Reiter, 1987; Felfernig et al., 2012) are a general proposal (explanation) for resolving an inconsistency. (3) *How* explanations are more related to specific aspects of a recommendation process, for example, when using a utility-based approach for item ranking (Felfernig et al., 2006), explanations can take into account corresponding weights to explain how a recommendation has been determined, for example, *since you have ranked the priority of the feature fps (frame rate per second) very high, item X is the one which is ranked highest since it received the highest utility value on the basis of our evaluation function*. *How* explanations play a major role in the context of group recommendations—specifically to achieve goals such as fairness and consensus (Tran et al., 2023; Felfernig et al., 2024b). An example of taking into account such aspects is the following: *since the preferences of user X have not been taken into account for already 3 months, the current restaurant recommendation specifically focuses on taking into account the preferences of X*.

5 Research directions

Basic insights from our analysis are the following. A major strength of knowledge-based (specifically constraint-based) recommenders is their ability to enforce domain-specific constraints. These systems are useful when recommending and explaining high-involvement items such as cars and financial services. A major disadvantage is "setup" efforts that are needed to predefine the recommendation knowledge in terms of product properties, product catalogs, similarity metrics, and constraints. There are many new developments in *knowledge-based (recommender) systems* focusing on a "deep" integration with *machine learning*, thus helping to unify the two worlds (Popescu et al., 2022). However, there are still a couple of research directions that are of interest—see the following discussion.

5.1 Integrating knowledge-based systems with machine learning

Specifically in constraint-based recommender systems, a major line of research focuses on the integration of concepts and techniques from machine learning into constraint-based reasoning, for example, by learning variable and variable value ordering heuristics (Popescu et al., 2022; Uta et al., 2022). This way, recommendation functionalities can be directly integrated with constraint solving. An issue for future research is to analyze alternative integrations, for example, in which way constraint-based reasoning can be integrated into machine learning—such a goal can be achieved using neuro-symbolic AI concepts (Monroe, 2022).

5.2 Diagnosis performance optimization

A major runtime performance bottleneck in knowledge-based recommenders are conflict detection and diagnosis algorithms which operate on the basis of individual consistency checks (Jannach, 2006). A related topic for future research is to intensively integrate machine learning techniques that can help to reduce the number of irrelevant consistency checks as much as possible and thus help to improve runtime performance of conflict detection and diagnosis.

5.3 Cognitive aspects of preference elicitation

Human decision making is amenable to different types of biases with a potential negative impact on decision quality (Atas et al., 2021). Recommender user interfaces and corresponding preference elicitation dialogs have to take this aspect into account. A major issue for future research is to understand in detail in which way recommender user interfaces can have an impact on decision processes and how to counteract decision biases in such contexts.

5.4 Evaluation metrics for knowledge-based recommenders

Current approaches to evaluate recommender systems primarily focus on the aspect of prediction quality and user acceptance (Uta et al., 2021). However, recommender systems also have a huge impact on how customers perceive items, which items are selected, and—as a consequence—which items have to be produced. This way, recommender systems have to be evaluated with regard to various additional aspects such as the visibility of items in recommendation lists (e.g., how often is an item highly-ranked) and restrictiveness of specific features in recommendation dialogs (e.g., are there too few items that support specific feature combinations). Initial related work on these aspects can be found in the study by Lubos et al. (2023).

5.5 Consequence-based explanations

Explanations in recommender systems focus on reasons as to why specific items have been recommended, why no solution could be identified, and how items have been determined (Friedrich and Zanker, 2011; Tintarev and Masthoff, 2012). Future research should focus more on the generation of explanations that take into account

TABLE 18 Examples of LLMs in knowledge-based digital camera recommendation.

Scenario	Example (prompt)	LLM response (excerpt)
Flexibly explaining physical properties	Why does a lower-resolution camera often provide a higher image quality?	In some cases, lower-resolution cameras have larger individual pixels. Larger pixels can capture more light, which can contribute to better image quality, especially in low-light conditions.
Flexible item comparisons	What is the most important difference between the Canon EOS 5d mark IV and the 5d mark II in terms of image quality?	The Mark IV has a wider ISO range and generally better low-light performance compared to the Mark II. The increased resolution does not sacrifice sensitivity, and the Mark IV is capable of producing cleaner images at higher ISO settings.

The examples already show the high flexibility in providing explanations to recommender users.

the consequences of choosing specific items, for example, in the context of investment decisions, a consequence-based explanation could be of type *since you prefer to purchase a quite expensive car, this will have an impact on both, the quality of living (in terms of having a smaller house) and in terms of being able to have money for holidays*.

5.6 Integration of large language models

The application of large language models (LLMs) in knowledge-based recommender systems (and beyond) starts to play a major role in the context of various tasks such as preference elicitation (e.g., in terms of building semantically-enriched user models) and explanations, for example, by identifying the best explanation strategy in specific recommendation contexts (see Table 18—the examples have been generated with ChatGPT 3.5).[11]

5.7 Sustainability aspects of recommender systems

Recommender systems can play a major role in achieving individual sustainability goals. These systems can help, for example, to systematically optimize energy consumption strategies in households, optimize resource reuse in supply chains, and improve personal wellbeing (Murphy et al., 2015; Felfernig et al., 2023b,c). A major issue for future research is to more intensively integrate psychological approaches such as nudging (Thaler and Sunstein, 2021) into recommendation dialogs and to develop new algorithms which help to increase the selection share of sustainable items.

11 openai.com

5.8 Generalizability to other knowledge-based systems

The concepts discussed in this article can be applied to other systems beyond knowledge-based recommender systems. For example, the concepts of personalized constraint-based recommendation and corresponding diagnosis concepts are also applicable in the context of configuration and re-configuration for highly-variant and complex products and services and also in the context of scheduling and re-scheduling (Felfernig et al., 2014).

6 Conclusion

With this article, we provide an overview of major concepts of the knowledge-based recommender systems extending existing overviews with new technological developments. In this context, we provide working examples that help to understand the underlying techniques and algorithms in more detail. Finally, as a result of our literature analysis, we discuss different research directions in knowledge-based recommendation which will help to stimulate further related research.

Author contributions

MU: Conceptualization, Data curation, Formal analysis, Investigation, Project administration, Software, Validation, Writing – original draft, Writing – review & editing. AF: Conceptualization, Methodology, Supervision, Validation, Writing – original draft, Writing – review & editing. V-ML: Conceptualization, Validation, Writing – review & editing. TT: Validation, Writing – review & editing. DG: Validation, Writing – review & editing. SL: Validation, Writing – review & editing. TB: Validation, Writing – review & editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. This work was supported by TU Graz Open Access Publishing Fund.

Conflict of interest

MU was employed by Siemens Energy AG.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Aggarwal, C. C. (2016). *Recommender Systems: The Textbook, 1st Edn*. Springer Nature Switzerland AG.

Almalis, N., Tsihrintzis, G., and Kyritsis, E. (2018). A constraint-based job recommender system integrating fodra. *Int. J. Comp. Intell. Stud.* 7, 103–123. doi: 10.1504/IJCISTUDIES.2018.094894

Atas, M., Felfernig, A., Polat-Erdeniz, S., Popescu, A., Tran, T. N. T., and Uta, M. (2021). Towards psychology-aware preference construction in recommender systems: overview and research issues. *J. Intell. Inf. Syst.* 57, 467–489. doi: 10.1007/s10844-021-00674-5

Bahramian, Z., and Ali Abbaspour, R. (2015). An ontology-based tourism recommender system based on spreading activation model. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* 40, 83–90. doi: 10.5194/isprsarchives-XL-1-W5-83-2015

Biere, A., Heule, M., Maaren, H., and Walsh, T. (2021). *Handbook of Satisfiability, 2nd Edn*. Amsterdam: IOS Press.

Bokolo, A. (2021). A case-based reasoning recommender system for sustainable smart city development. *AI Soc.* 36, 159–183. doi: 10.1007/s00146-020-00984-2

Bouraga, S., Jureta, I., Faulkner, S., and Herssens, C. (2014). Knowledge-based recommendation systems: a survey. *Int. J. Intell. Inf. Technol.* 10, 1–19. doi: 10.4018/ijiit.2014040101

Bridge, D., Göker, M. H., McGinty, L., and Smyth, B. (2005). Case-based recommender systems. *Knowl. Eng. Rev.* 20, 315–320. doi: 10.1017/S0269888906000567

Bridge, D. G. (2002). "Towards conversational recommender systems: a dialogue grammar approach," in *ECCBR Workshops* (Aberdeen), 9–22.

Burke, R. (2000). Knowledge-based recommender systems. *Encyclop. Libr. Inf. Syst.* 69, 180–200. Available online at: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=dc133144d431fc3b75c8de27f6bb21da6eb5bc1b

Burke, R. (2002). Hybrid recommender systems: survey and experiments. *User Model. User Adapt. Interact.* 12, 331–370. doi: 10.1023/A:1021240730564

Burke, R., Felfernig, A., and Göker, M. H. (2011). Recommender systems: an overview. *AI Mag.* 32, 13–18. doi: 10.1609/aimag.v32i3.2361

Burke, R., Hammond, K., and Young, B. (1996). Knowledge-based navigation of complex information spaces. *Natl. Conf. Artif. Intell.* 462, 468.

Cena, F., Console, L., and Vernero, F. (2021). Logical foundations of knowledge-based recommender systems: a unifying spectrum of alternatives. *Inf. Sci.* 546, 60–73. doi: 10.1016/j.ins.2020.07.075

Chen, L., and Pu, P. (2012). Critiquing-based recommenders: survey and emerging trends. *User Model User Adapt. Interact.* 22, 125–150. doi: 10.1007/s11257-011-9108-6

Chen, L., Yan, D., and Wang, F. (2017). User perception of sentiment-integrated critiquing in recommender systems. *Int. J. Hum. Comput. Stud.* 121, 4–20. doi: 10.1016/j.ijhcs.2017.09.005

Christakopoulou, K., Radlinski, F., and Hofmann, K. (2016). "Towards conversational recommender systems," in *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16* (New York, NY: ACM), 815–824.

Colombo-Mendoza, L. O., Valencia-García, R., Rodríguez-González, A., Alor-Hernández, G., and Samper-Zapater, J. J. (2015). RecomMetz: a context-aware knowledge-based mobile recommender system for movie showtimes. *Expert Syst. Appl.* 42, 1202–1222. doi: 10.1016/j.eswa.2014.09.016

Cordero, P., Enciso, M., López, D., and Mora, A. (2020). A conversational recommender system for diagnosis using fuzzy rules. *Expert Syst. Appl.* 154:113449. doi: 10.1016/j.eswa.2020.113449

Cöster, R., Gustavsson, A., Olsson, T., and Rudström, Å. (2002). "Enhancing web-based configuration with recommendations and cluster-based help," in *AH'2002*

Workshop on Recommendation and Personalization in eCommerce, 28 May 2002 (Málaga), 1–10.

Dai, S., Shao, N., Zhao, H., Yu, W., Si, Z., Xu, C., et al. (2023). "Uncovering chatgpt?s capabilities in recommender systems," in *17th ACM Conference on Recommender Systems, RecSys '23* (New York, NY: Association for Computing Machinery), 1126–1132.

Daoudi, A., Mechqrane, Y., Bessiere, C., Lazaar, N., and Bouyakhf, E. H. (2016). "Constraint acquisition using recommendation queries," in *International Joint Conferences on Artificial Intelligence* (Palo Alto, CA), 720–726.

Dara, S., Chowdary, C. R., and Kumar, C. (2020). A survey on group recommender systems. *J. Intell. Inf. Syst.* 54, 271–295. doi: 10.1007/s10844-018-0542-3

Dong, M., Zeng, X., Koehl, L., and Zhang, J. (2020). An interactive knowledge-based recommender system for fashion product design in the big data environment. *Inf. Sci.* 540, 469–488. doi: 10.1016/j.ins.2020.05.094

Dragone, P., Teso, S., and Passerini, A. (2018). Constructive preference elicitation. *Front. Robot. AI* 4:71. doi: 10.3389/frobt.2017.00071

Eiter, T., Ianni, G., and Krennwallner, T. (2009). "Answer set programming: a primer," in *Reasoning Web Semantic Technologies for Information Systems: International Summer School*, eds S. Tessaris, E. Franconi, T. Eiter, C. Gutierrez, S. Handschuh, M.-C. Rousset, and R. A. Schmidt (Berlin Heidelberg: Springer), 40–110. doi: 10.1007/978-3-642-03754-2_2

Ekstrand, M., Riedl, J., and Konstan, J. (2011). Collaborative filtering recommender systems. *Found. Trends Human Comp. Interact.* 4. doi: 10.1561/1100000009

Erdeniz, S. P., Felfernig, A., and Atas, M. (2022). Applying matrix factorization to consistency-based direct diagnosis. *Appl. Intell.* 52, 7024–7036. doi: 10.1007/s10489-020-02183-4

Erdeniz, S. P., Felfernig, A., Samer, R., and Atas, M. (2019). "Matrix factorization based heuristics for constraint-based recommenders," in *34th ACM/SIGAPP Symposium on Applied Computing* (New York, NY: ACM), 1655–1662.

Esheiba, L., Elgammal, A., Helal, I. M., and El-Sharkawi, M. E. (2021). A hybrid knowledge-based recommender for product-service systems mass customization. *Information* 12:296. doi: 10.3390/info12080296

Falkner, A., Felfernig, A., and Haag, A. (2011). Recommendation technologies for configurable products. *AI Mag.* 32, 99–108. doi: 10.1609/aimag.v32i3.2369

Fano, A., and Kurth, S. W. (2003). "Personal choice point: Helping users visualize what it means to buy a bmw," in *8th International Conference on Intelligent User Interfaces, IUI '03* (New York, NY: Association for Computing Machinery), 46–52.

Fargier, H., Gimenez, P.-F., and Mengin, J. (2016). "Recommendation for product configuration: an experimental evaluation," in *18th International Configuration Workshop (CWS 2016) within CP 2016: 22nd International Conference on Principles and Practice of Constraint Programming* (Albi: École des Mines d'Albi-Carmaux), 9.

Feely, C., Caulfield, B., Lawlor, A., and Smyth, B. (2020). "Using case-based reasoning to predict marathon performance and recommend tailored training plans," in *Case-Based Reasoning Research and Development: 28th International Conference, ICCBR 2020, Salamanca, Spain, June 8-12, 2020, Proceedings 28* (Springer Nature Switzerland AG), 67–81.

Felfernig, A. (2007). Standardized configuration knowledge representations as technological foundation for mass customization. *IEEE Transact. Eng. Manag.* 54, 41–56. doi: 10.1109/TEM.2006.889066

Felfernig, A., Atas, M., Stettinger, M., Tran, T., and Reiterer, S. (2024a). "Group recommender applications," in *Group Recommender Systems: An Introduction*, eds A. Felfernig, L. Boratto, M. Stettinger, and M. Tkalcic (Springer), 79–94.

Felfernig, A., Boratto, L., Stettinger, M., and Tkalcic, M. (2024b). *Group Recommender Systems: An Introduction*. Springer Publishing Company Inc.

Felfernig, A., and Burke, R. (2008). "Constraint-based recommender systems: technologies and research issues," in *10th International Conference on Electronic Commerce, ICEC '08* (New York, NY: Association for Computing Machinery), 1–10.

Felfernig, A., Friedrich, G., Isak, K., Shchekotykhin, K., Teppan, E., and Jannach, D. (2009a). Automated debugging of recommender user interface descriptions. *Appl. Intell.* 31, 1–14. doi: 10.1007/s10489-007-0105-8

Felfernig, A., Friedrich, G., Jannach, D., and Zanker, M. (2006). An integrated environment for the development of knowledge-based recommender applications. *Int. J. Electron. Commerce* 11, 11–34. doi: 10.2753/JEC1086-4415110201

Felfernig, A., Friedrich, G., Jannach, D., and Zanker, M. (2011). "Developing constraint-based recommenders," in *Recommender Systems Handbook*, eds F. Ricci, L. Rokach, B. Shapira, and P. Kantor (Springer), 187–215.

Felfernig, A., Friedrich, G., Schubert, M., Mandl, M., Mairitsch, M., and Teppan, E. (2009b). "Plausible repairs for inconsistent requirements," in *21st International Joint Conference on Artificial Intelligence, IJCAI'09* (San Francisco, CA. Morgan Kaufmann Publishers Inc.), 791–796.

Felfernig, A., Hotz, L., Bagley, C., and Tiihonen, J. (2014). *Knowledge-based Configuration: From Research to Business Cases*. San Francisco, CA: Morgan Kaufmann Publishers Inc.

Felfernig, A., Isak, K., Szabo, K., and Zachar, P. (2007). "The VITA financial services sales support environment," in *19th National Conference on Innovative Applications of Artificial Intelligence - Volume 2, IAAI'07* (AAAI Press), 1692–1699.

Felfernig, A., and Kiener, A. (2005). "Knowledge-based interactive selling of financial services with FSAdvisor," in *17th Conference on Innovative Applications of Artificial Intelligence - Volume 3, IAAI'05* (AAAI Press), 1475–1482.

Felfernig, A., Le, V., and Lubos, S. (2023a). "Conjunctive query based constraint solving for feature model configuration," in *12th Conference on Information Technology and Its Applications (CITA 2023)*, *volume 734 of Lecture Notes in Networks and Systems* (Springer), 357–367.

Felfernig, A., Le, V.-M., Popescu, A., Uta, M., Tran, T. N. T., and Atas, M. (2021). "An overview of recommender systems and machine learning in feature modeling and configuration," in *15th International Working Conference on Variability Modelling of Software-Intensive Systems, VaMoS '21* (ACM), 1–8.

Felfernig, A., Reiterer, S., Stettinger, M., Reinfrank, F., Jeran, M., and Ninaus, G. (2013a). "Recommender systems for configuration knowledge engineering," in *15th International Configuration Workshop, ConfWS '13*, 51–54.

Felfernig, A., Reiterer, S., Stettinger, M., and Tiihonen, J. (2015). "Towards understanding cognitive aspects of configuration knowledge formalization," in *9th International Workshop on Variability Modelling of Software-Intensive Systems, VaMoS '15* (New York, NY: Association for Computing Machinery), 117–123.

Felfernig, A., Schippel, S., Leitner, G., Reinfrank, F., Isak, K., Mandl, M., et al. (2013b). Automated repair of scoring rules in constraint-based recommender systems. *AI Commun.* 26, 15–27. doi: 10.3233/AIC-120543

Felfernig, A., Schubert, M., and Reiterer, S. (2013c). "Personalized diagnosis for over-constrained problems," in *23rd International Joint Conf. on Artificial Intelligence, IJCAI '13* (AAAI), 1990–1996.

Felfernig, A., Schubert, M., and Zehentner, C. (2012). An efficient diagnosis algorithm for inconsistent constraint sets. *AI EDAM* 26, 53–62. doi: 10.1017/S0890060411000011

Felfernig, A., Stettinger, M., Atas, M., Samer, R., Nerlich, J., Scholz, S., et al. (2018a). "Towards utility-based prioritization of requirements in open source environments," in *2018 IEEE 26th International Requirements Engineering Conference (RE)* (IEEE), 406–411.

Felfernig, A., Walter, R., Galindo, J., Benavides, D., Atas, M., Polat-Erdeniz, S., et al. (2018b). Anytime diagnosis for reconfiguration. *J. Intell. Inf. Syst.* 51, 161–182. doi: 10.1007/s10844-017-0492-1

Felfernig, A., Wundara, M., Tran, T., Le, V.-M., Lubos, S., and Polat-Erdeniz, S. (2023b). Sports recommender systems: overview and research issues. *arXiv.* doi: 10.21203/rs.3.rs-3710874/v1

Felfernig, A., Wundara, M., Tran, T. N. T., Polat-Erdeniz, S., Lubos, S., El Mansi, M., et al. (2023c). Recommender systems for sustainability: overview and research issues. *Front. Big Data* 6:1284511. doi: 10.3389/fdata.2023.1284511

Fesenmaier, D., Ricci, F., Schaumlechner, E., Wöber, K., and Zanella, C. (2003). "Dietorecs: travel advisory for multiple decision styles," in *Information and Communication Technologies in Tourism 2003*, eds A. Frew, M. Hitz, and P. O?Connors (New York, NY: Springer), 232–241.

Friedrich, G., and Zanker, M. (2011). A taxonomy for generating explanations in recommender systems. *AI Mag.* 32, 90–98. doi: 10.1609/aimag.v32i3.2365

Gao, C., Lei, W., He, X., de Rijke, M., and Chua, T.-S. (2021). Advances and challenges in conversational recommender systems: a survey. *AI Open* 2, 100–126. doi: 10.1016/j.aiopen.2021.06.002

Goeker, M., and Thompson, C. (2000). "Personalized conversational case-based recommendation," in *Advances in Case-Based Reasoning, Volume 1898 of Lecture Notes in Computer Science*, eds E. Blanzieri, and L. Portinale (Berlin; Heidelberg: Springer), 709–714.

Grasch, P., Felfernig, A., and Reinfrank, F. (2013). "ReComment: towards critiquing-based recommendation with speech interaction," in *7th ACM Conference on Recommender Systems, RecSys '13* (New York, N: ACM), 157–164.

Güell, M., Salamó, M., Contreras, D., and Boratto, L. (2020). Integrating a cognitive assistant within a critique-based recommender system. *Cogn. Syst. Res.* 64:1–14. doi: 10.1016/j.cogsys.2020.07.003

Hernandez-Nieves, E., Hernández, G., Gil-Gonzalez, A. B., Rodríguez-González, S., and Corchado, J. M. (2021). Cebra: a case-based reasoning application to recommend banking products. *Eng. Appl. Artif. Intell.* 104:104327. doi: 10.1016/j.engappai.2021.104327

Iaquinta, L., deGemmis, M., Lops, P., Semeraro, G., Filannino, M., and Molino, P. (2008). "Introducing serendipity in a content-based recommender system," in *8th International Conference on Hybrid Intelligent Systems, HIS '08* (Barcelona: IEEE Computer Society), 168–173.

Jannach, D. (2006). "Techniques for fast query relaxation in content-based recommender systems," in *29th Annual German Conference on Artificial Intelligence* (Berlin, Heidelberg: Springer), 49–63.

Jannach, D., and Kreutler, G. (2007). Rapid development of knowledge-based conversational recommender applications with advisor suite. *J. Web Eng.* 6, 165–192. Available online at: https://www.researchgate.net/profile/Dietmar-Jannach/publication/220538223_Rapid_Development_of_Knowledge-Based_Conversational_Recommender_Applications_with_Advisor_Suite/links/5e30386a299bf1cdb9f69472/Rapid-Development-of-Knowledge-Based-Conversational-Recommender-Applications-with-Advisor-Suite.pdf?_sg%5B0%5D=started_experiment_milestone&origin=journalDetail

Jannach, D., Manzoor, A., Cai, W., and Chen, L. (2021). A survey on conversational recommender systems. *ACM Comp. Surv.* 54, 1–36. doi: 10.1145/3453154

Junker, U. (2004). "QuickXPlain: Preferred explanations and relaxations for over-constrained problems," in *19th National Conference on Artifical Intelligence, AAAI'04* (AAAI Press), 167–172.

Khan, A., and Hoffmann, A. (2003). Building a case-based diet recommendation system without a knowledge engineer. *Artif. Intell. Med.* 27, 155–179. doi: 10.1016/S0933-3657(02)00113-6

Knijnenburg, B. P., Willemsen, M. C., Gantner, Z., Soncu, H., and Newell, C. (2012). Explaining the user experience of recommender systems. *User Model. User Adapt. Interact.* 22, 441–504. doi: 10.1007/s11257-011-9118-4

Kolodner, J. (2014). *Case-Based Reasoning*. San Mateo, CA: Morgan Kaufmann.

Kolodner, J. L. (1992). An introduction to case-based reasoning. *Artif. Intell. Rev.* 6, 3–34. doi: 10.1007/BF00155578

Law, E., and von Ahn, L. (2011). *Human Computation, 1st Edn*. Kentfield, CA: Morgan & Claypool Publishers.

Le, V., Felfernig, A., Uta, M., Benavides, D., Galindo, J., and Tran, T. (2023). "FASTDIAGP: an algorithm for parallelized direct diagnosis," in *AAAI Conference on Artificial Intelligence*, 6442–6449.

Le, V.-M., Felfernig, A., Uta, M., Benavides, D., Galindo, J., and Tran, T. N. T. (2021). "DirectDebug: automated testing and debugging of feature models," in *43rd International Conference on Software Engineering: New Ideas and Emerging Results, ICSE-NIER '21* (Madrid: IEEE Press), 81–85.

Le, V.-M., Tran, T. N. T., and Felfernig, A. (2022). "Consistency-based integration of multi-stakeholder recommender systems with feature model configuration," in *26th ACM International Systems and Software Product Line Conference-Volume B*, 178–182.

Lee, H. J., and Kim, H. S. (2015). "eHealth recommendation service system using ontology and case-based reasoning," in *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)* (Chengdu: IEEE), 1108–1113.

Lee, T., Chun, J., Shim, J., and Lee, S. (2006). An ontology-based product recommender system for B2B marketplaces. *Int. J. Electron. Commerce* 11, 125–155. doi: 10.2753/JEC1086-4415110206

Lika, B., Kolomvatsos, K., and Hadjiefthymiades, S. (2014). Facing the cold start problem in recommender systems. *Expert Syst. Appl.* 41, 2065–2073. doi: 10.1016/j.eswa.2013.09.005

Lorenzi, F., and Ricci, F. (2005). "Case-based recommender systems: a unifying view," in *Intelligent Techniques for Web Personalization, Volume 3169 of Lecture Notes in Computer Science*, eds B. Mobasher and S. Anand (Berlin; Heidelberg: Springer), 89–113.

Lubos, S., Le, V.-M., Felfernig, A., and Tran, T. N. T. (2023). "Analysis operations for constraint-based recommender systems," in *17th ACM Conference on Recommender Systems, RecSys '23* (New York, NY: ACM), 709–714.

Maccatrozzo, V., Terstall, M., Aroyo, L., and Schreiber, G. (2017). "Sirup: serendipity in recommendations via user perceptions," in *22nd International Conference on Intelligent User Interfaces, IUI '17* (New York, NY: ACM), 35–44.

Mandl, M., and Felfernig, A. (2012). "Improving the performance of unit critiquing," in *20th International Conference on User Modeling, Adaptation, and Personalization, UMAP'12* (Berlin, Heidelberg: Springer-Verlag), 176–187.

Masthoff, J. (2015). "Group recommender systems: aggregation, satisfaction and group attributes," in *Recommender Systems Handbook, Chapter 22* (New York, NY: Springer Science+Business Media), 743–776.

McCarthy, K., McGinty, L., Smyth, B., and Salam,ó, M. (2006). "The needs of the many: a case-based group recommender system," in *ECCBR 2006, Volume 4106 of Lecture Notes in Computer Science*, eds T. Roth-Berghofer, M. Göker, and H. Güvenir (Berlin; Heidelberg: Springer), 196–210.

McCarthy, K., Reilly, J., McGinty, L., and Smyth, B. (2005). "Experiments in dynamic critiquing," in *10th International Conference on Intelligent User Interfaces, IUI '05* (New York, NY: Association for Computing Machinery), 175–182.

McCarthy, K., Salem, Y., and Smyth, B. (2010). "Experience-based critiquing: reusing critiquing experiences to improve conversational recommendation," in *ICCBR'10* (Springer), 480–494.

McGuinness, D. (2007). "Configuration," in *The Description Logic Handbook: Theory, Implementation and Applications, 2 Edn* (Cambridge: Cambridge University Press), 417–435.

McSherry, D. (2003). "Similarity and compromise," in *Case-Based Reasoning Research and Development*, eds K. D. Ashley, and D. G. Bridge (Berlin, Heidelberg: Springer Berlin Heidelberg), 291–305.

Mirzadeh, N., Ricci, F., and Bansal, M. (2005). "Feature selection methods for conversational recommender systems," in *2005 IEEE International Conference on E-Technology, e-Commerce and e-Service (EEE'05) on e-Technology, e-Commerce and e-Service, EEE '05* (Hong Kong: IEEE Computer Society), 772–777.

Monroe, D. (2022). Neurosymbolic AI. *Commun. ACM* 65, 11–13. doi: 10.1145/3554918

Murphy, S. Ó., Manzano, Ó., and Brown, K. N. (2015). "Design and evaluation of a constraint-based energy saving and scheduling recommender system," in *International Conference on Principles and Practice of Constraint Programming* (Springer International Publishing Switzerland), 687–703.

Murti, Y., A., Baizal, Z., and Adiwijaya, K. (2016). Compound critiquing for conversational recommender system based on functional requirement. *Adv. Sci. Lett.* 22, 1892–1896. doi: 10.1166/asl.2016.7770

Musto, C., Semeraro, G., Lops, P., De Gemmis, M., and Lekkas, G. (2015). Personalized finance advisory through case-based recommender systems and diversification strategies. *Decis. Support Syst.* 77, 100–111. doi: 10.1016/j.dss.2015.06.001

Pazzani, M. J., and Billsus, D. (2007). "Content-based recommendation systems," in *The Adaptive Web: Methods and Strategies of Web Personalization* (Berlin; Heidelberg: Springer), 325–341.

Pereira, J., Matuszyk, P., Krieter, S., Spiliopoulou, M., and Saake, G. (2016). A feature-based personalized recommender system for product-line configuration. *SIGPLAN Not.* 52:2993249. doi: 10.1145/2993236.2993249

Pessemier, T. D., Dhondt, J., and Martens, L. (2017). Hybrid group recommendations for a travel service. *Multimed. Tools Appl.* 76, 2787–2811. doi: 10.1007/s11042-016-3265-x

Popescu, A., Polat-Erdeniz, S., Felfernig, A., Uta, M., Atas, M., Le, V.-M., et al. (2022). An overview of machine learning techniques in constraint solving. *J. Intell. Inf. Syst.* 58, 91–118. doi: 10.1007/s10844-021-00666-5

Pramod, D., and Bafna, P. (2022). Conversational recommender systems techniques, tools, acceptance, and adoption: a state of the art review. *Expert Syst. Appl.* 203:117539. doi: 10.1016/j.eswa.2022.117539

Rafter, R., and Smyth, B. (2005). Conversational collaborative recommendation—An experimental analysis. *Artif. Intell. Rev.* 24, 301–318. doi: 10.1007/s10462-005-9004-8

Reilly, J., McCarthy, K., McGinty, L., and Smyth, B. (2004). "Dynamic critiquing," in *Advances in Case-Based Reasoning: 7th European Conference, ECCBR 2004, Madrid, Spain, August 30-September 2, 2004. Proceedings 7* (Berlin; Heidelberg: Springer), 763–777.

Reiter, R. (1987). A theory of diagnosis from first principles. *Artif. Intell.* 32, 57–95. doi: 10.1016/0004-3702(87)90062-2

Rossi, F., van Beek, P., and Walsh, T. (2006). *Handbook of Constraint Programming*. Amsterdam: Elsevier Science.

Sha, X., Sun, Z., and Zhang, J. (2021). Hierarchical attentive knowledge graph embedding for personalized recommendation. *Electron. Commer. Res. Appl.* 48:101071. doi: 10.1016/j.elerap.2021.101071

Smyth, B., McGinty, L., Reilly, J., and McCarthy, K. (2004). "Compound critiques for conversational recommender systems," in *IEEE/WIC/ACM International Conference on Web Intelligence (WI'04)* (Beijing: IEEE), 145–151.

Sun, R., Cao, X., Zhao, Y., Wan, J., Zhou, K., Zhang, F., et al. (2020). "Multi-modal knowledge graphs for recommender systems," in *29th ACM International Conference on Information & Knowledge Management*, 1405–1414.

Tazl, O. A., Perko, A., and Wotawa, F. (2019). "Conversational recommendations using model-based reasoning," in *CEUR Workshop Proceedings, Vol. 2467* (Otzenhausen: RWTH Aachen), 13–19.

Temple, P., Acher, M., Jézéquel, J.-M., and Barais, O. (2017). Learning contextual-variability models. *IEEE Softw.* 34, 64–70. doi: 10.1109/MS.2017.4121211

Teppan, E., and Zanker, M. (2020). "Exploiting answer set programming for building explainable recommendations," in *ISMIS 2020, volume 12117 of Lecture Notes in Computer Science* (Springer), 395–404.

Thaler, R., and Sunstein, C. (2021). *Nudge: The Final Edition*. New Haven, CT; London: Yale University Press.

Tintarev, N., and Masthoff, J. (2012). Evaluating the effectiveness of explanations for recommender systems. *User Model. User Adapt. Interact.* 22, 399–439. doi: 10.1007/s11257-011-9117-5

Towle, B., and Quinn, C. (2000). "Knowledge based recommender systems using explicit user models," in *AAAI Technical Report WS-00-04* (AAAI Press), 74–77.

Tran, T., Felfernig, A., and Le, V. (2023). An overview of consensus models for group decision-making and group recommender systems. *User Model. User Adapt. Interact*. doi: 10.1007/s11257-023-09380-z

Ulz, T., Schwarz, M., Felfernig, A., Haas, S., Shehadeh, A., Reiterer, S., et al. (2017). Human computation for constraint-based recommenders. *J. Intell. Inf. Syst*. 49, 37–57. doi: 10.1007/s10844-016-0433-4

Uta, M., Felfernig, A., Helic, D., and Le, V.-M. (2022). "Accuracy- and consistency-aware recommendation of configuratiosn," in *26th ACM International Systems and Software Product Line Conference - Volume A, SPLC '22* (New York, NY: Association for Computing Machinery), 79–84.

Uta, M., Felfernig, A., Le, V., Popescu, A., Tran, T., and Helic, D. (2021). "Evaluating recommender systems in feature model configuration," in *25th ACM International Systems and Software Product Line Conference - Volume A, SPLC '21* (New York, NY: Association for Computing Machinery), 58–63.

Walter, R., Felfernig, A., and Küchlin, W. (2017). Constraint-based and SAT-based diagnosis of automotive configuration problems. *J. Intell. Inf. Syst*. 49, 87–118. doi: 10.1007/s10844-016-0422-7

Wang, H., Zhang, F., Zhang, M., Leskovec, J., Zhao, M., Li, W., et al. (2019a). "Knowledge-aware graph neural networks with label smoothness regularization for recommender systems," in *25th ACM SIGKDD International Conference on Knowledge Discovery* & *Data Mining* (New York, NY: ACM), 968–977.

Wang, H., Zhao, M., Xie, X., Li, W., and Guo, M. (2019b). "Knowledge graph convolutional networks for recommender systems," in *The World Wide Web Conference* (New York, NY: ACM), 3307–3313.

Weckesser, M., Kluge, R., Pfannemüller, M., Matthé, M., Schürr, A., and Becker, C. (2018). "Optimal reconfiguration of dynamic software product lines based on performance-influence models," in *22nd International Systems and Software Product Line Conference, Vol. 1* (New York, NY: ACM), 98–109.

Wobcke, W., Krzywicki, A., Kim, Y. S., Cai, X., Bain, M., Compton, P., et al. (2015). A deployed people-to-people recommender system in online dating. *AI Mag*. 36, 5–18. doi: 10.1609/aimag.v36i3.2599

Wohlin, C. (2014). "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *18th Intl. Conference on Evaluation and Assessment in Software Engineering* (New York, NY: ACM), 1–10.

Wu, G., Luo, K., Sanner, S., and Soh, H. (2019). "Deep language-based critiquing for recommender systems," in *13th ACM Conference on Recommender Systems, RecSys '19* (New York, NY: Association for Computing Machinery), 137–145.

Xie, H., Wang, D. D., Rao, Y., Wong, T.-L., Raymond, L. Y., Chen, L., et al. (2018). Incorporating user experience into critiquing-based recommender systems: a collaborative approach based on compound critiquing. *Int. J. Mach. Learn. Cybernet*. 9, 837–852. doi: 10.1007/s13042-016-0611-2

Zanker, M., Bricman, M., Gordea, S., Jannach, D., and Jessenitschnig, M. (2006). "Persuasive online-selling in quality and taste domains," in *7th International Conference on E-Commerce and Web Technologies, EC-Web'06* (Berlin, Heidelberg: Springer-Verlag), 51–60.

Zhang, J., Jones, N., and Pu, P. (2008). "A visual interface for critiquing-based recommender systems," in *9th ACM Conference on Electronic Commerce* (New York, NY: ACM), 230–239.

Zhou, K., Zhao, W. X., Bian, S., Zhou, Y., Wen, J.-R., and Yu, J. (2020). "Improving conversational recommender systems via knowledge graph based semantic fusion," in *26th ACM SIGKDD International Conference on Knowledge Discovery* & *Data Mining* 1006–1014.

Zhu, Q., Zhou, X., Wu, J., Tan, J., and Guo, L. (2020). "A knowledge-aware attentional reasoning network for recommendation," in *AAAI Conference on Artificial Intelligence* (Palo Alto, CA: AAAI), 6999–7006.

Ziarani, R., and Ravanmehr, R. (2021). Serendipity in recommender systems: a systematic literature review. *J. Comput. Sci. Technol*. 36, 375–396. doi: 10.1007/s11390-020-0135-9

Zou, J., Chen, Y., and Kanoulas, E. (2020). "Towards question-based recommender systems," in *43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '20* (New York, NY: ACM), 881–890.