11

Case-Based Recommendation

Barry Smyth[1,2]

[1] The School of Computer Science and Informatics,
University College Dublin, Belfield, Dublin 4, Ireland
[2] ChangingWorlds Ltd.
South County Business Park, Leopardstown,
Dublin 18, Ireland.
Barry.Smyth@ucd.ie

Abstract. Recommender systems try to help users access complex information spaces. A good example is when they are used to help users to access online product catalogs, where recommender systems have proven to be especially useful for making product suggestions in response to evolving user needs and preferences. Case-based recommendation is a form of content-based recommendation that is well suited to many product recommendation domains where individual products are described in terms of a well defined set of features (e.g., *price*, *colour*, *make*, etc.). These representations allow case-based recommenders to make judgments about product similarities in order to improve the quality of their recommendations and as a result this type of approach has proven to be very successful in many e-commerce settings, especially when the needs and preferences of users are ill-defined, as they often are. In this chapter we will describe the basic approach to case-based recommendation, highlighting how it differs from other recommendation technologies, and introducing some recent advances that have led to more powerful and flexible recommender systems.

11.1 Introduction

Recently I wanted to buy a new digital camera. I had a vague idea of what I wanted—a 6 mega-pixel digital SLR from a good manufacturer—but it proved difficult and time consuming to locate a product online that suited my needs, especially as these needs evolved during my investigations. Many online stores allowed me to *browse* or *navigate* through their product catalog by choosing from a series of static features (e.g., *manufacturer, camera type, resolution, level of zoom* etc.). Each time I selected a feature I was presented with the set of cameras with this feature and I could then go on to choose another feature to further refine the presented products. Other stores allowed me to *search* for my ideal camera by entering a query (e.g. *"digital slr, 6 mega-pixels"*) and presented me with a list of results which I could then browse at my leisure.

Both of these access options were helpful in different ways—in the beginning I preferred to browse through catalogs but, after getting a feel for the various features and compromises, I tended to use search-based interfaces—however neither provided

P. Brusilovsky, A. Kobsa, and W. Nejdl (Eds.): The Adaptive Web, LNCS 4321, pp. 342–376, 2007.

me with the flexibility I really sought. For a start, all of the stores I tried tended to slavishly respect my queries. This was especially noticeable when no results could be returned to satisfy my stated needs; this is often referred to as *stonewalling* [17]. For instance, looking for a 6 mega-pixel digital SLR for under $200 proved fruitless—unsurprising perhaps to those 'in the know'—and left me with no choice but to start my search again. This was especially frustrating when there were many cameras that were similar enough to my query to merit suggestion. Moreover, stonewalling is further compounded by a *diversity problem*: I was frequently presented with sets of products that were all very similar to each other thus failing to offer me a good set of alternatives. At other times I would notice a camera that was almost perfect, aside from perhaps one or two features, but it was usually difficult to provide this form of feedback directly. This *feedback problem* prevented me from requesting *"another camera like this one but with more optical zoom and/or a lower price"*, for instance.

In all, perhaps one of the most frustrating aspects of my search was the apparent inability of most online stores to learn anything about my preferences over time. In my opinion shopping for an expensive item such as a digital camera is an exercise in patience and deliberation, and one that is likely to involve many return visits to particular online stores. Unfortunately, despite the fact that I had spent a significant time and effort searching and browsing for cameras during previous visits none of the stores I visited had any facility to remember my previous interactions or preferences. For instance, my reluctance to purchase a very expensive camera—I never accepted recommendations for cameras above $1000—should have been recognised and factored into the store's recommendations, but it was not. As a result many of my interactions turned out to be requests for less expensive suggestions. This *preference problem* meant that starting my searches from scratch became a regular feature of these visits.

Recommender systems are designed to address many of the problems mentioned above, and more besides, by offering users a more intelligent approach to navigating and searching complex information spaces. They have been especially useful in many e-commerce domains with many stores using recommendation technologies to help convert browsers into buyers by providing intelligent and timely sales support and product suggestions; see for example Chapter 16 of this book [38] for a survey of recommendation techniques in an e-commerce setting. One of the key features of many recommendation technologies is the ability to consider the needs and preferences of the individual when it comes to generating *personalized* recommendations or suggestions. We will return to this issue later in this chapter but also refer the interested reader to related work on the development of personalization technologies. For example, Chapters 2 [35] and 4 [44] of this book consider different approaches to learning and modeling the preferences of users while Chapters 3 [62], 6 [61], and 18 [8] of this book consider different ways in which user models may be harnessed to provide users with more personalized access to online information and services. Indeed, while many recommendation and personalization technologies focus on the needs of the individual, some researchers have begun to consider group recommendation scenarios where the potentially competing preferences of a number of individuals need to be considered; see for example Chapter 20 [41] of this book.

Recommendation techniques come in two basic flavours. *Collaborative filtering* approaches rely on the availability of user ratings information (e.g. *"John likes items A, B and C but dislikes items E and F"* and make suggestions for a target user based on the items that similar users have liked in the past, without relying on any information about the items themselves other than their ratings; see Chapter 9 [83] of this book for a more detailed account of collaborative filtering approaches. In contrast *content-based* techniques rely on item descriptions and generate recommendations from items that are similar to those the target user has liked in the past, without directly relying on the preferences of other users; see Chapter 10 [69] of this book for a detailed account of pure content-based approaches.

Case-based recommenders implement a particular style of content-based recommendation that is very well suited to many product recommendation scenarios; see also [16]. They rely on items or products being represented in a structured way using a well defined set of features and feature values; for instance, in a travel recommender a particular vacation might be presented in terms of its *price, duration, accommodation, location, mode of transport, etc.* In turn the availability of similarity knowledge makes it possible for case-based recommenders to make fine-grained judgments about the similarities between items and queries for informing high-quality suggestions to the user. Case-based recommender systems are the subject of this chapter, where we will draw on a range of examples from a variety of recommender systems, both research prototypes and deployed applications. We will explain their origins in case-based reasoning research [1, 31, 46, 101] and their basic mode of operation as recommender systems. In particular, we will look at how case-based recommenders deal with the issues highlighted above in terms of their approach to selection similarity, recommendation diversity, and the provision of flexible feedback options. In addition we will consider the use of case-based recommendation techniques to produce suggestions that are personalized for the needs of the individual user and in this way present case-based approaches as one important solution for Web personalization problems; see also Chapters 2 [35], 3 [62], and 16 [38] in this book for related work in the area of Web personalization.

11.2 Towards Case-Based Recommendation

Case-based recommender systems have their origins in *case-based reasoning* (CBR) techniques [1, 46, 101, 48, 99]. Early case-based reasoning systems were used in a variety of problem solving and classification tasks and can be distinguished from more traditional problem solving techniques by their reliance on concrete experiences instead of problem solving knowledge in the form of codified rules and strong domain models. Case-based reasoning systems rely on a database (or case base) of past problem solving experiences as their primary source of problem-solving expertise. Each case is typically made up of a *specification* part, which describes the problem at hand, and a *solution* part, which describes the solution used to solve this problem. New problems are solved by retrieving a case whose specification is similar to the current target problem and then adapting its solution to fit the target situation. For example, CLAVIER [39] is a case-based reasoning system used by Lockheed to assist in determining the layout of materials to be cured in an autoclave (i.e., a large convection oven used, in this

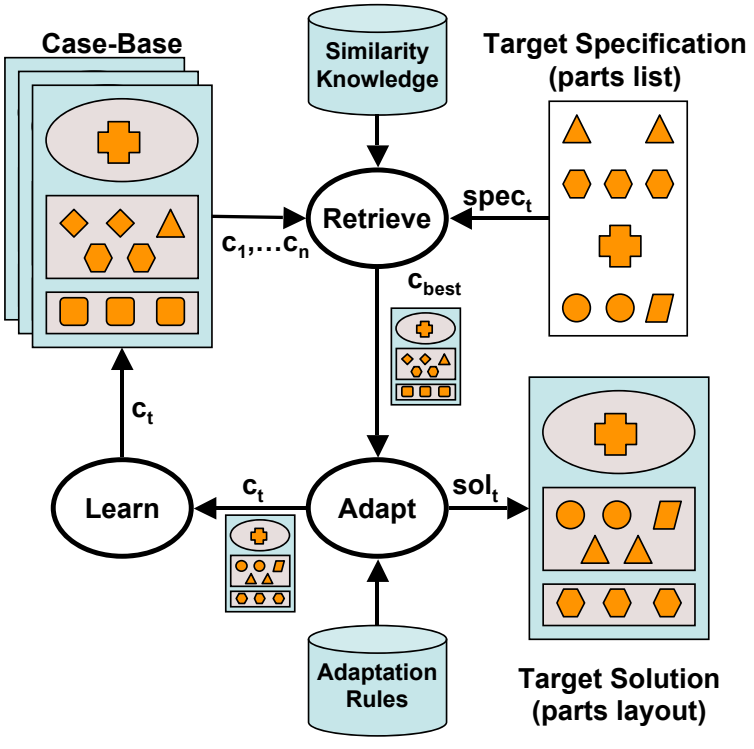

Fig. 11.1. CLAVIER uses CBR to design layout configurations for a set of parts to be cured in an autoclave. This is a complex layout task that does not lend itself to a traditional knowledge-based approach. However a case base of high-quality past layouts can be readily assembled. New layouts for a target parts-list can then be produced by retrieving a case with a similar parts-list and adapting its layout. If successful this new layout can then be learned by storing it in the case base as a new case.

case, for the curing of composite materials for aerospace applications). CLAVIER has the job of designing a good layout—one that will maximise autoclave throughput—for a new parts-list. The rules for determining a good layout are not well understood but previous layouts that have proved to be successful are readily available. CLAVIER uses these previous layout examples as the cases in its case base. Each case is made up of a parts-list (its specification) and the particular layout used (its solution). New layouts for a new parts-list are determined by matching the new parts-list against these cases and adapting the layout solution used by the most similar case; see Figure 11.1. CLAVIER has been a huge practical success and has been in use for a number of years by Lockheed, virtually eliminating the production of low-quality parts that must be scrapped, and saving thousands of dollars each month.

Case-based recommenders borrow heavily from the core concepts of retrieval and similarity in case-based reasoning. Items or products are represented as cases and recommendations are generated by retrieving those cases that are most similar to a user's

query or profile. The simplest form of case-based recommendation is presented in Figure 11.2. In this figure we use the example of a digital camera recommender system, with the product case base made up of detailed descriptions of individual digital cameras. When the user submits a target query—in this instance providing a relatively vague description of their requirements in relation to *camera price* and *pixel resolution*—they are presented with a ranked list of k recommendations which represent the top k most similar cases that match the target query. As a form of content-based recommendation

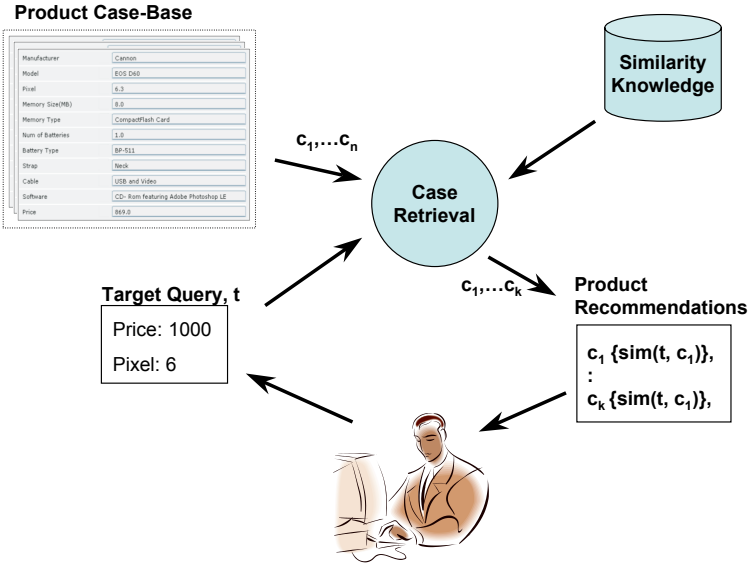

Fig. 11.2. In its simplest form a case-based recommendation system will retrieve and rank product suggestions by comparing the user's target query to the descriptions of products stored in its case base using similarity knowledge to identify products that are close matches to the target query.

(see, for example, [5, 26, 63, 78, 94] and also Chapter 10 [69] of this book) case-based recommenders generate their recommendations by looking to the item descriptions, with items suggested because they have similar descriptions to the user's query. There are two important ways in which case-based recommender systems can be distinguished from other types of content-based systems: (1) the manner in which products are represented; and (2) the way in which product similarity is assessed. Both of these will be discussed in detail in the following sections.

11.2.1 Case Representation

Normally content-based recommender systems operate in situations where content items are represented in an unstructured or semi-structured manner. For example, the NewsDude content-recommender, which recommends news articles to users, assumes

text-based news stories and leverages a range of keyword-based content analysis techniques during recommendation; see for example, [9] and Chapter 18 [8] in this book. In contrast, case-based recommender systems rely on more structured representations of item content. These representations are similar to those used to represent case-knowledge in case-based reasoners. For example, they often use a set of well-defined features and feature values to describe items, rather than free-form text. This reliance on structured content means that case-based recommenders are particularly well adapted to many consumer recommendation domains, particularly e-commerce domains, where detailed feature-based product descriptions are often readily available.

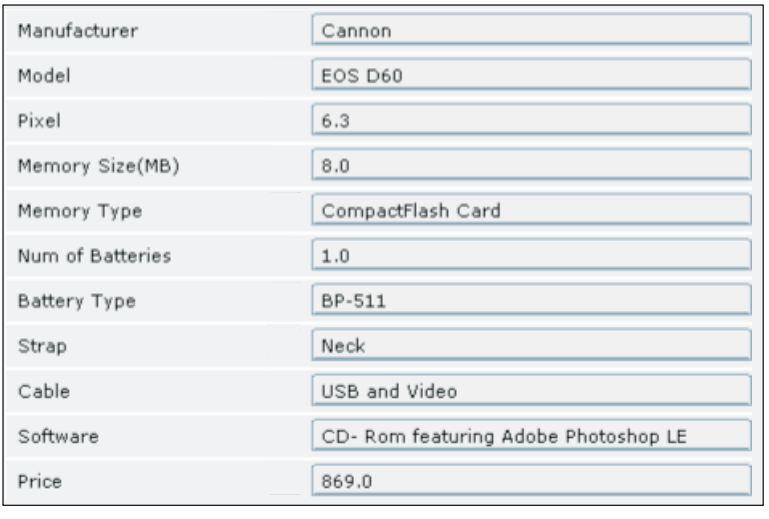

Fig. 11.3. An example product case from a digital camera product catalog.

Figure 11.3 shows one such example product case from a catalog of cameras. The case is for a *Canon* digital camera and, as can be seen, the product details are captured using 11 different features (e.g., *manufacturer, model, memory type, price, etc.*) with each feature associated with one of a well-defined space of possible feature values (e.g., the *manufacturer* feature values are drawn from a well-defined set of possible manufacturers such as *Canon, Nikon, Sony etc.*). The example also highlights how different types of features can be used within a product description. In this case, a mixture of *numeric* and *nominal* features are used. For instance, *price* is an example of a numeric feature, which obviously represents the cost of the camera, and can take on values anywhere in a range of possible prices, from about $100 to upwards of $3000. Alternatively, *memory type* is a nominal feature, whose values come from a well-defined set of alternatives corresponding to the 4 or 5 different memory card options that are commonly used by digital cameras. The Entree recommender is another good example of a case-based recommender system. This system will be explored in more detail in Section 11.4 but suffice it to say that Entree is designed to make restaurant suggestions; see Figure 11.4. In terms of its core representation, Entree also uses a structured case format—although

Fig. 11.4. Entree [21, 22] recommends restaurants to users based on a variety of features such as *price, cuisine type, atmosphere,* etc..

the presentation in Figure 11.12 is largely textual the basic case representation is fundamentally feature-based—using features such as *price, cuisine type, atmosphere,* etc. to represent each restaurant case.

11.2.2 Similarity Assessment

The second important distinguishing feature of case-based recommender systems relates to their use of various sophisticated approaches to similarity assessment when it comes to judging which cases to retrieve in response to some user query. Because case-based recommenders rely on structured case representations they can take advantage of more structured approaches to similarity assessment than their content-based cousins. For example, traditional content-based techniques tend to use keyword-based similarity metrics, measuring the similarity between a user query and a product in terms of the frequency of occurrence of overlapping query terms within the product text. If the user is looking for a *"$1000 6 mega-pixel DSLR"* then cameras with all of these terms will be rated highly, and depending on the strength of the similarity criterion used, if no cameras exist with all of these terms then none may be retrieved. We have already highlighted this type of retrieval inflexibility (stonewalling) as a critical problem in the introduction to this chapter. Any reasonable person would be happy to receive a recommendation for a *"$900 6.2 mega-pixel digital SLR"*, for the above query, even though strictly speaking there is no overlap between the terms used in this description and the query.

Case-based recommenders can avail of more sophisticated similarity metrics that are based on an explicit mapping of case features and the availability of specialised feature level similarity knowledge. An online property recommender might use case-

based techniques to make suggestions that are similar to a target query even when exact matches are not available. For example, a user who looking for a *"2 bedroom apartment in Dublin with a rent of 1150 euro"* might receive recommendations for properties that match the *bedroom* feature and that are *similar* to the target query in terms of *price* and *location*; the recommendations might offer slightly higher or lower priced properties in a nearby location when no exact matches are available.

$$Similarity(t,c) = \frac{\sum_{i=1..n} w_i * sim_i(t_i, c_i)}{\sum_{i=1..n} w_i} \tag{11.1}$$

Assessing similarity at the case level (or between the target query and a candidate case) obviously involves combining the individual feature level similarities for the relevant features. The usual approach is to use a weighted sum metric such as that shown in Equation 11.1. In brief, the similarity between some target query, t and some candidate case (or item), c, is the weighted sum of the individual similarities between the corresponding features of t and c, namely t_i and c_i. Each weight encodes the relative importance of a particular feature in the similarity assessment process and each individual feature similarity is calculated according to a similarity function that is defined for that feature, $sim_i(t_i, c_i)$. For instance, looking to the property recommender example above, if *rent* is very important to the user then the weight associated with this feature will be higher than the weights associated with less important features. In turn, when it comes to comparing the query and a case in terms of their *rent* the recommender system may draw on a specialised similarity metric designed for comparing monthly rents. A different metric might be used for comparing the *number of bedrooms* or the *property type*.

We must also consider the source of the individual feature level similarities and how they can be calculated. For example, returning to our camera recommender system, consider a numeric feature such as *pixel resolution*. The target query and a candidate case might be compared in terms of this feature using a similarity metric with the sort of similarity profile shown in Figure 11.5(a); maximum similarity is achieved when the *pixel resolution* of a candidate case matches that of the target query, and for cases with higher or lower *pixel resolution* there is a corresponding decline in similarity. This is an example of a *symmetric* similarity metric because there is no bias in favour of either higher or lower resolution cases.

$$sim_{price}(p_t, p_c) = 1 - \frac{|p_t - p_c|}{max(p_t, p_c)} \tag{11.2}$$

Sometimes symmetric similarity metrics are not appropriate. For instance, consider the *price* feature: it is reasonable to expect that a user will view cameras with prices (p_c) that are lower than their target price (p_t) to be preferable to cameras with higher prices, all other things being equal. The similarity metric in Equation 11.2 is used in many recommender systems (e.g., see [52, 75]) as one way to capture this notion and the metric displays a similarity profile similar to that shown in Figure 11.5(b). For instance, consider a $1000 target price and two candidate cases, one with a price of $500 and one for $1500. In terms of the symmetric similarity metric represented by Figure 11.5(a), the latter candidate corresponds to the point x in Figure 11.5(a) and the former to point y.

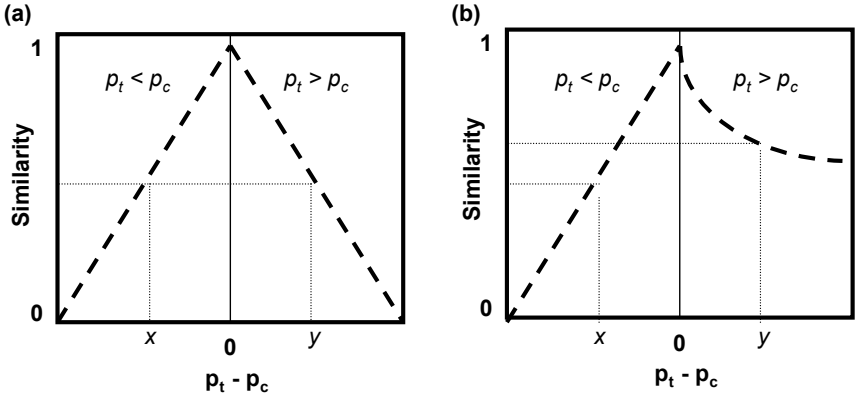

Fig. 11.5. Two example similarity profiles for numeric similarity metrics: (a) corresponds to a standard symmetric similarity metric; (b) corresponds to an asymmetric metric that gives preference to features values that are lower than the target's value.

For both cases the similarity assessment is the same, reflecting that both differ from the target price by the same amount, $500, with no preference given to whether a case is less or more expensive. These cases are plotted in the same way in Figure 11.5(b) but now we see that the more expensive case (point *x*) has a lower similarity than the cheaper camera (point *y*). Even though both candidates differ by $500, preference is given to the cheaper case.

To evaluate the similarity of non-numeric features in a meaningful way requires additional domain knowledge. For example, in a vacation recommender it might be important to be able to judge the similarities of cases of different *vacation types*. Is a *skiing* holiday more similar to a *walking* holiday than it is to a *city break* or a *beach* holiday? One way to make such judgments is by referring to suitable domain knowledge such as an ontology of vacation types. In Figure 11.6 we present part of what such an ontology might look like with different feature values represented as nodes and similar feature values grouped near to each other. In this way, the similarity between two arbitrary nodes can be evaluated as an inverse function of the distance between them or the distance to their nearest common ancestor. Accordingly, a *skiing* holiday is more similar to a *walking* holiday (they share a direct ancestor, *activity* holidays) than it is to a *beach* holiday, where the closest common ancestor is the ontology root node.

11.2.3 Acquiring Similarity Knowledge

Similarity assessment is obviously a key issue for case-based reasoning and case-based recommender systems. Of course the availability and use of similarity knowledge (feature-based similarity measures and weighting functions) is an important distinguishing feature of case-based recommendation. Although further detailed discussion of this particular issue is beyond the scope of this chapter, it is nonetheless worth considering the origin of this knowledge in many systems. For the most part this knowledge is hand-coded: similarity tables and trees, such as the vacation ontology above, are made

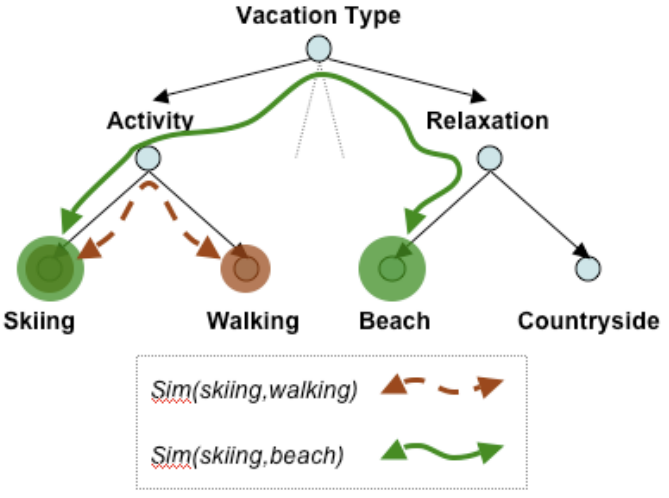

Fig. 11.6. A partial ontology of vacation types can be used as the basis for similarity judgments for non-numeric features.

available and codified by a domain knowledge expert. Similarly, importance weights might be assigned by the user at retrieval time or by the domain expert during system design. Hand-coding this knowledge is, of course, expensive and so increasingly researchers have begun to explore how machine learning techniques can be used to relieve these knowledge acquisition costs.

A number of researchers have looked at the issue of automatically learning the feature weights that are be used to influence the level of importance of different features during the case similarity calculations. Wettschereck and Aha [102], for example, describe an evaluation of a number of weight-learning algorithms that are *knowledge-poor*, in the sense that they avoid the need for detailed domain knowledge to drive the learning process. They show how even these knowledge-poor techniques can result in significant improvements in case-based classification tasks and present a general framework for understanding and evaluating different weight-learning approaches; see also the work of [42, 81] for approaches to local weight-learning in CBR based on reinforcement learning.

Stahl [96] also looks at feature-weight learning but describes an alternative approach in which feedback is provided by a "similarity teacher" whose job it is to evaluate the ordering a given retrieval set. For example, in a recommender context a user may play the role of the similarity teacher because her selections can be interpreted as retrieval feedback; if our user selects product number 3 in the list of recommendations first then we can conclude that the correct ordering should have placed this product at the top of the list. Stahl's learning algorithm attempts to minimise the average ordering error in retrieval sets. The work of Cohen et al. [25] looks at the related issue of learning to order items given feedback in the form of preference judgements. They describe a technique for automatically learning a preference function to judge how advisable it is to rank some item i ahead of item j, and go on to show how a set of items can then

be ranked by attempting to maximise agreements with this learned preference function; see also the work of [13].

Finally, it is worth highlighting recent similarity learning work by O'Sullivan et al. [66, 67, 68]. This work has not been used directly by case-based recommenders but instead has been used to improve the quality of collaborative filtering recommenders (see Chapter 9 [83] in this book) by using case-based style similarity metrics when evaluating profile similarities. Normally a collaborative filtering recommender system can only evaluate the similarity between two profiles if they share ratings. For example in a TV recommender two users that have both rated *ER* and *Frasier* can be compared. But if one user has only rated *ER* and the other has only rated *Frasier* then they cannot be compared. O'Sullivan et al. point out that the ratings patterns within a collaborative filtering database can be analysed to estimate the similarity between programmes like *ER* and *Frasier*. They show that by using data-mining techniques it is possible to discover that, for example, 60% of the people who have liked *ER* have also liked *Fraiser*, and use this as a proxy for the similarity between these two programmes. They demonstrate how significant improvements in recommendation accuracy can be obtained by using these similarity estimates with more sophisticated case-based profile similarity metrics.

11.2.4 Single-Shot Recommendation

Many case-based recommenders operate in a *reactive* and *single-shot* fashion, presenting users with a single set of recommendations based on some initial query; thus the user is engaged in a single (short-lived) interaction with the system. For example, the Analog Devices OpAmp recommender presents a user with a set of available OpAmps that closely match the user's query [100, 103]. The online property recommender referred to earlier operate similarly, responding with a selection of suitable apartments in response to a user's rental constraints; see also the *DubLet* system by [40].

The point to make here is that single-shot recommendation has its shortcomings. In particular, if users do not find what they are looking for among the initial recommendations—as is frequently the case—then their only option is to revise their query and start again. Indeed the pure similarity-based nature of most case-based recommender systems increases the chances of this happening in certain situations because, as we discussed earlier, the top ranked recommendations may differ from the target query in more or less the same ways. As a result they will be very similar to each other—they will lack diversity—and if the user doesn't like the first recommendation she is unlikely to be satisfied with the similar alternatives either. In the remaining sections of this chapter we will explore how this simple model of case-based recommendation has been extended to provide a more sophisticated recommendation framework, one that provides for more sophisticated interaction between recommender and user, generating personalized recommendations that are more diverse, through an extended dialog with the user.

11.3 Similarity and Beyond

Let us look at a concrete example of the diversity problem referred to above. Consider a vacation recommender where a user submits a query for a 2-week vacation for two in the sun, costing less than $750, within 3 hours flying time of Ireland, and with good night-life and recreation facilities on-site. The top recommendation returned is for an apartment in the Hercules complex in the Costa Del Sol, Spain, for the first two weeks in July. A good recommendation by all accounts, but what if the second, third, and fourth recommendations are from the same apartment block, albeit perhaps for different two-week periods during the summer, or perhaps for different styles of apartments? While the k ($k = 4$ in this case) best recommendations are all very similar to the target query, they are also very similar to each other. The user has not received a useful set of alternatives if the first recommendation is unsuitable. This scenario is not uncommon

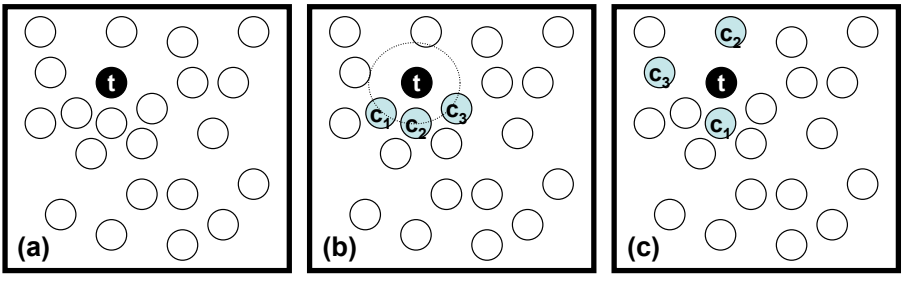

Fig. 11.7. Similarity vs diversity during case retrieval: (a) a case base with highlighted target query, t; (b) a conventional similarity-based retrieval strategy returns the cases that are individually closest to the target query, thus limiting their potential diversity; (c) an alternative retrieval strategy that balances similarity to the target and the relative diversity of the selected cases produces a more diverse set of recommendations.

in recommender systems that employ similarity-based retrieval strategies: they often produce recommendation sets that lack diversity and thus limit user options (see Figure 11.7(a&b)). These observations have led a number of researchers to explore alternatives to similarity-based retrieval, alternatives that attempt to explicitly improve recommendation diversity while at the same time maintaining query similarity. [1]

11.3.1 Similarity vs. Diversity

How then can we improve the diversity of a set of recommended cases, especially since many of the more obvious approaches are likely to reduce the similarity of the selected

[1] Incidentally, related concerns regarding the primacy of similarity in other forms of case-based reasoning have also come to light, inspiring many researchers to look for alternative ways to judge the *utility* of a case in a given problem solving context (e.g. [7, 19, 34, 45, 49, 91]). For example, researchers have looked at the importance of adaptability alongside similarity, arguing that while a case may appear to be similar to a target problem, this does not mean it can be successfully adapted for this target (see [49, 91]).

cases compared to the target query? In case-based recommenders, which implement a similarity-based retrieval strategy, the trade-off between similarity and diversity is often straightforward when we look at the similarity and diversity characteristics for the top k items. For low values of k, while similarity to the target query tends to be high, the diversity between the top k recommendations tends to be very low. In other words, the top ranking cases are often similar to the target query in more or less the same ways; of course what we really need is a set of recommendations that are equally similar to the target query but in different ways. As we move through the top ranking recommendations we tend to find cases that are similar to the target query but increasingly different from those that have gone before. These are the interesting cases to consider from a recommendation diversity perspective.

We attempt to capture this visually in Figure 11.8(a) by depicting a list of the top 9 recommendations in decreasing order of their similarity to the target query. Each recommendation is shaded to reflect its diversity relative to the others. For example, the top 3 recommendations are all shaded to the same degree, indicating that the are all very similar to each other. Hence there is little variation among the top 3 results; perhaps these are all examples of vacation suggestions for the same Spanish apartment complex for the first few weeks of July. It should be clear in this example how more diverse recommendations only begin to appear for higher values of k. Recommendations 4, 6 and 9, for example, are more diverse alternatives; perhaps these suggestions correspond to vacations in Tuscany or on the Costa Brava. One solution then is to look for ways of identifying and promoting these more diverse recommendations so that the user is presented with a more diverse list of suggestions, which still remain true to their target query. Figure 11.8(b) illustrates this: recommendations from positions 4,6 and 9 in Figure 11.8(a) are promoted to positions 2, 3 and 4 (or perhaps the less diverse suggestions of 2, 3, and 5 are simply removed from the suggestion list) thus providing the user with variation in the top recommendations.

One way to improve diversity is to simply select k random cases from the top bk most similar cases to the target query. This so-called *bounded random selection* strategy was proposed by [92] but it was shown to result in an unacceptable drop in query similarity, and so is hardly practical in many recommendation scenarios. However, more principled approaches are available, which rely on an explicit model of diversity. We can define the diversity of a set of retrieved cases, $c_1, ...c_k$, to be the average *dissimilarity* between all pairs of these cases (Equation 11.3). Then the *bounded greedy selection* strategy proposed by [92] offers a way to improve diversity, while at the same time maintaining target query similarity; see also [11]. This strategy incrementally builds a diverse retrieval set of k cases, R, by starting from a set of the bk most similar cases to the target query. During each step the remaining cases are ordered according to their *quality* with the highest quality case added to R. The key to this algorithm is a quality metric that combines diversity and similarity (Equation 11.4). The quality of a case c is proportional to the similarity between c and the current target t, and to the diversity of c *relative* to those cases so far selected, $R = \{r_1, ..., r_m\}$; see Equation 11.5. The first case to be selected is always the one with the highest similarity to the target. However during subsequent iterations, the case selected is the one with the highest combination of similarity to the target and diversity with respect to the set of cases selected so far.

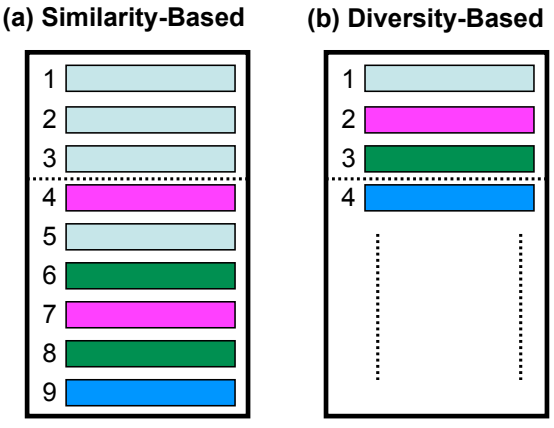

Fig. 11.8. Typical approaches to similarity-based recommendation tend to produce recommendation lists with limited diversity characteristics, such as the list shown in (a). Individual items are shaded to reflect their diversity characteristics so that in (a) items 1,2,3 and 5 are very similar to each other as are items 4 and 7 and items 6 and 8. In (b) a different ordering of items is presented, one that maximises the diversity of the top items.

$$Diversity(c_1,...c_n) = \frac{\sum_{i=1..n}\sum_{j=i..n}(1 - Similarity(c_i,c_j))}{\frac{n}{2} * (n-1)} \tag{11.3}$$

```
1.   define BoundedGreedySelection (t, C, k, b)
2.   begin
3.     C' := bk cases in C that are most similar to t
4.     R := {}
5.     For i := 1 to k
6.       Sort C' by Quality(t,c,R) for each c in C'
7.       R  := R + First(C')
8.       C' := C' - First(C')
9.     EndFor
10.  return R
11.  end
```

Fig. 11.9. The Bounded Greedy Selection strategy for producing a diverse set of k: t refers to the current target query; C refers to the case base; k is the size of the desired retrieval/recommendation set; b refers to the bound used for the initial similarity-based retrieval.

$$Quality(t,c,R) = Similarity(t,c) * RelDiversity(c,R) \tag{11.4}$$

$$\begin{aligned} RelDiversity(c,R) &= 1 \; if \; R = \{\}; \\ &= \frac{\sum_{i=1..m}(1 - Similarity(c,r_i))}{m}, otherwise \end{aligned} \tag{11.5}$$

Empirical studies presented in [11, 92] demonstrate the diversity benefits of the above approach. In particular, the bounded greedy algorithm is found to provide a cost effective diversity enhancing solution, resulting in significant improvements in recommendation diversity against relatively minor reductions in target query similarity. For example, [11, 92] applied the technique in a number of different recommender systems including a vacation recommender and a recruitment advisor. In the vacation recommender, examining the similarity and diversity characteristics of the top 3 recommendations reveals that the bounded greedy technique manages to achieve a 50% improvement in relative diversity when compared to the standard similarity-based recommendation approach but suffers a minor loss of less than 10% in similarity to the target query. Similar results are found in the recruitment domain and in practice users are seen to benefit from a much more varied selection of alternatives that remain similar to their stated needs.

11.3.2 Alternative Diversity-Preserving Approaches

The *bounded greedy* technique discussed above was among the first practical attempts to explicitly enhance the diversity of a set of recommendations without significantly compromising their query similarity characteristics; although it is worth noting that some loss of similarity is experienced with this approach. In parallel Shimazu [86, 87] introduced an alternative method for enhancing the diversity of a set of recommendations. In brief, a set of 3 recommendations, c_1, c_2 and c_3, are chosen relative to some query q such that c_1 is maximally similar to q, c_2 is maximally dissimilar to c_1 and then c_3 is maximally dissimilar to c_1 and c_2. In this way, the triple of cases are chosen to be maximally diverse but, unlike the bounded greedy technique above, the similarity of c_2 and c_3 to the query is likely to be compromised. As such the value of this approach is limited to situations where the set of recommended cases is drawn from a set of cases that are all sufficiently similar to the user query to begin with.

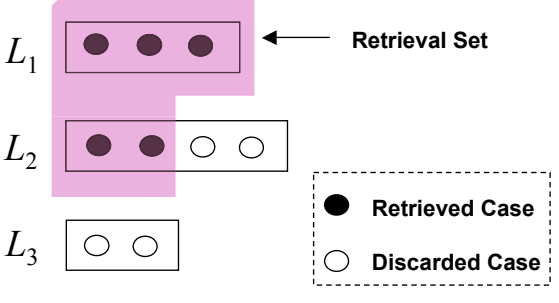

Fig. 11.10. The approach described in [55, 56] partitions the case base into similarity layers—groups of cases with equivalent similarity to the target query—and the retrieved cases are chosen starting with the highest similarity layer and until k cases have been selected. The final cases selected from the lowest necessary similarity layer are chosen based on an optimal diversity maximizing technique.

Recently a number of alternative diversity enhancing selection techniques have been proposed. For example, [55] shows that it is sometimes possible to enhance diversity without loss of query similarity and a related approach based on the idea of *similarity layers* is described [56]. Very briefly, a set of cases, ranked by their similarity to the target query are partitioned into similarity layers, such that all cases in a given layer have the same similarity value to the query. To select a set of k diverse cases, the lowest similarity layer that contributes cases to the recommendation set is identified and a subset of cases from this layer are selected for inclusion in the final recommended set with all cases in higher similarity layers automatically included; see Figure 11.10. Cases are selected from this lowest similarity layer using an optimal diversity maximizing algorithm. This approach has the ability to improve diversity while at the same time fully preserving the similarity of cases to the user query. However, the diversity improvements obtained are typically less than those achieved by the bounded greedy algorithm, because all cases from higher similarity layers are always included without any diversity enhancement. An alternative, and more flexible, diversity enhancing approach is also introduced based on the analogous notion of *similarity intervals*; see also [56]. The advantage of this approach is that it can achieve greater diversity improvements by relaxing the constraint that query similarity must be preserved. Query similarity is reduced but within a tolerance level defined by the width of the similarity intervals.

It is also worth noting that a retrieval technique may not be designed to explicitly enhance diversity but may nonetheless have a beneficial effect by its very nature. *Order-based retrieval* is a good example of such a technique [14, 17]. It is based on the idea that the relative similarities of cases to a query of *ideal* feature values is one way of ordering a set of cases for recommendation. Order-based retrieval constructs an ordering relation from the query provided by the user and applies this relation to the case base of products returning the k items at the top of the ordering. The order relation is constructed from the composition of a set of canonical operators for constructing partial orders based on the feature types that make up the user query. While the technical details of order-based retrieval are beyond the scope of this chapter the essential point to note is that an empirical evaluation of order-based retrieval demonstrates that it has an inherent ability to enhance the diversity of a set of retrieval results; that is, the cases at the top of the ordering tend to be more diverse than an equivalent set of cases ranked based on their pure similarity to the user query.

In [58] McSherry proposes a *compromise-driven* approach to retrieval in recommender systems. This approach is inspired by the observation that the most similar cases to the user's query are often not representative of compromises that the user may be prepared to accept. Compromise-driven retrieval is based on a variation of the usual similarity assumption: that a given case is more acceptable than another if it is more similar to the user's query *and* it involves a subset of the compromises that the other case involves. As well as being less likely to be contradicted by user behaviour, this assumption serves as the basis for a more principled approach to deciding which cases are included in the retrieval set than setting an arbitrary similarity threshold over the candidate cases. For example, no case is included in the retrieval set if there is a more similar case that involves a subset of the compromises it involves. Though not relying explicitly on diversity as an additional measure of recommendation quality, compromise-driven

retrieval does offer users a better (usually more diverse) set of recommendation alternatives. Moreover, the recommendation set is guaranteed to provide full coverage of the available cases in the sense that for any case that is not included in the retrieval set, one of the recommended cases is at least as good in terms of its similarity to the user's query and the compromises it involves. While the size of the retrieval set required to provide full coverage cannot be predicted in advance, experimental results suggest that retrieval-set sizes tend to remain within reasonable limits even for queries of realistic complexity; see [58].

In summary then, we have seen how recent developments in case-based recommendation have relaxed the conventional wisdom of the similarity assumption, in favour of retrieval strategies that are more likely to deliver a recommendation set that offers users a more diverse set of alternatives. In the next section will will revisit the diversity issue in a slightly different context. While accepting the value of diversity during recommendation, we will question whether it should always be used as a retrieval constraint or whether there are occasions when it is more useful to focus on similarity or diversity.

11.4 The Power of Conversation

As mentioned earlier, the single-shot model of recommendation, whether similarity-based or diversity-enhanced, is limited to a single interaction between the user and the recommender system. If the user is not satisfied with the recommendations they receive then their only option is to modify their query and try again. Indeed the single-shot approach also makes the assumption that the user is in a position to provide a detailed query from the start, an assumption that does not often hold in practice. For example, in many product recommendation scenarios users may start with an initial query, which they will ultimately come to adapt and refine as they learn more about a particular product-space or the compromises that might be possible in relation to certain product features. Sometimes a user will come to disregard features that, initially at least, were important, as they recognise the value of other features, for example. These observations have motivated the development of conversational recommender systems which engage the user in an extended, interactive recommendation dialog during which time they attempt to elicit additional query information in order to refine recommendations[2]. Today most case-based recommenders employ conversational techniques, engaging users in an extended dialog with the system in order to help them navigate through a complex product space by eliminating items from consideration as a result of user feedback; note that these dialogs are normally restricted in the form of feedback solicited from the user, rather than offering free form natural language style dialogs.

Two different forms of conversational recommender systems can be distinguished according to the type of feedback that they solicit from users. In the nomenclature of Shimazu [86, 87], conversational recommenders can adopt a *navigation by asking* or a *navigation by proposing* style approach. In the case of the former, recommenders ask

[2] Conversational recommender systems have their origins in *conversational case-based reasoning (CCBR)* [3, 4, 12, 64], which apply similar techniques to elicit query information in problem solving domains and diagnostic tasks.

their users a series of questions regarding their requirements; this form of feedback is sometimes termed *value elicitation*. For example, a digital camera recommender might ask *"What style of camera do you want? Compact or SLR?"* or *"How much optical zoom do you need?"*. Alternatively, systems that employ navigation by proposing avoid posing direct questions in favour of presenting users with interim recommendations and asking for their feedback, usually in the form of a simple preference or a rating. Both styles of conversation have their pros and cons when it comes to user costs and recommendation benefits as we shall discuss in the following sections.

11.4.1 Navigation by Asking

Navigation by asking is undoubtedly the most direct way to elaborate a user's requirements and can lead to very efficient conversational dialogs in many situations. The Adaptive Place Advisor, which helps users to choose destinations such as restaurants, is good example of a system that employs navigation by asking [37, 98]. For instance, Figure 11.11 shows a sample Adaptive Place Advisor conversation between the user (the *inquirer*) and the recommender system (the *advisor*). Clearly, the Adaptive Place Advisor supports a sophisticated form of conversational dialog, employing natural language processing techniques in order to respond to freeform user answers. In addition, it is worth highlighting that its conversational form allows the user to ask questions of the recommender system too (see Line 3 of Figure 11.11), something that we will return to in Section 11.4.3, under the heading of mixed-initiative systems. The essential point is that each question that the Adaptive Place Advisor asks is designed to narrow down the possible recommendations that might be suggested to the user. Indeed in this system recommendations are only finally made when a few items remain.

One of the key issues that must be addressed with this style of recommendation concerns the important issue of what sets of questions should be asked during each session. For instance the order and number of questions can have a significant impact on usability and user effort. Doyle & Cunningham were among the first to report experimental results on this topic within CBR [32] by evaluating different question-selection criteria, including an entropy-based method which evaluates the informativeness or *information gain* of given features and is inspired by the decision-tree building work of [71, 73]; see also [86, 87]. Schmitt and his colleagues [84] propose an alternative approach, called *simVar*, that is especially tailored to e-commerce recommendation scenarios. Instead of using entropy-based information gain measures, this approach considers the influence that a given feature can have on the similarity of cases in the case base. Thus questions are asked of features that are deemed to have the highest influence on the similarity of cases. In addition, an estimate of the user's likely ability to answer such questions is also considered. Evaluations show that SimVar is capable of producing question dialogs that successfully prioritise questions that are both important and easy for the user to answer and that these can be more effective in product recommendation scenarios than those produced by entropy-based approaches; see also [6] for other useful dialog criteria. In Section 11.5 we will discuss how personalization techniques are used by the Adaptive Place Advisor to improve its conversational efficiency by eliminating the need to pose certain questions in situations where a learned model of the particular user can provide the answers.

1.	**Inquirer**:	*Where do you think I should eat tonight?*
2.	**Advisor**:	What type of food would you like?
3.	**Inquirer**:	*What types are there?*
4.	**Advisor**:	You can say things like Chinese, Indian, and Mediterranean.
5.	**Inquirer**:	*Oh, maybe a cheap Indian place.*
6.	**Advisor**:	What kind of parking would you like?
7.	**Inquirer**:	*I don't care, as long as it's in Palo Alto.*
8.	**Advisor**:	I'm sorry, I don't know of any restaurants like that, would you like to search for any price?
9.	**Inquirer**:	*No, I think I'd like Chinese instead.*
10.	**Advisor**:	How does Mandarin Gourmet on 420 Ramona sound?[1]
11.	**Inquirer**:	*No, what else do you have?*
12.	**Advisor**:	How about Jing-Jing Szechwan Hunan Gourmet on 443 Emerson Street?
13.	**Inquirer**:	*Sure, that sounds fine.*

Fig. 11.11. An example of the type of conversational dialog between a user (the *inquirer*) and the Adaptive Place Advisor recommender system (the *advisor*) in which the user is trying to decide on a restaurant for dinner.

The value elicitation approach adopted in navigation by asking has its origins in the classification and diagnosis tasks that are the focus of many conversational case-based reasoning systems (see for e.g., [3, 4, 12, 64]), but its use in recommendation tasks is not always appropriate. For example, users are often less inclined to tolerate protracted lists of direct questions. Oftentimes they will not know the answers to questions that demand a high-level of domain knowledge or they may reject questions that ask for sensitive or personal information. Moreover, providing answers to direct questions can impose a significant interfacing burden. For example, expecting users to respond with textual answers is not appropriate in the context of recommender systems that operate over mobile devices such as PDAs and mobile phones.

11.4.2 Navigation by Proposing

The above value elicitation issues have led to an increased interest in other forms of user feedback that are amenable to the navigation by proposing style of conversational recommendation. The key feature of navigation by proposing is that the user is presented with one of more recommendation alternatives, rather than a question, during each recommendation cycle, and they are invited to offer feedback in relation to these alternatives. In general terms there are 3 important types of feedback. *Ratings-based feedback* (see [43, 89, 90]) involves the user providing an explicit rating of a recommendation, but this form of feedback is more commonly found in collaborative filtering style recommender systems (see Chapter 9 [83] in this book) and shall not be discussed further here. Alternatively, feedback might be expressed in the form of a constraint over certain features of one of the recommendations (*critique-based feedback*)

Fig. 11.12. Entree [21, 22] recommends restaurants to users and solicits feedback in the form of feature *critiques*. The screenshot shows a recommendation for *Planet Hollywood* along with a variety of fixed critiques (*less$$, nicer, cuisine* etc.) over features such as the *price, ambiance* and *cuisine* of the restaurant.

[21, 22, 33, 50, 51, 70, 75]. Even simpler again is *preference-based feedback* in which the user expresses a preference for one alternative over the others [52].

Critique-Based Feedback. Critiquing-based recommenders allow users to provide feedback in the form of a directional feature constraint. For example, a digital camera shopper might ask for a camera that is *cheaper* than the current recommendation, *cheaper* being a critique over the *price* feature. The FindMe systems [21, 22] were among the first recommenders to champion critiquing as an effective form of feedback in conversational recommender systems. For instance, the Entree recommender suggests restaurants in Chicago and each recommendation allows the user to select from seven different critiques; see Figure 11.12. When a user selects a critique such as *cheaper*, Entree eliminates cases (restaurants) that do not satisfy the critique from consideration in the next cycle, and selects that case which is most similar to the current recommendation from those remaining; thus each critique acts as a filter over the cases. As a form of feedback, critiquing has much to recommend it. It proves to be an effective way to guide the recommendation process and yet provides users with a straightforward mechanism to provide feedback, one that requires limited domain knowledge on the part of the user and it is easy to implement with even the simplest of interfaces. The FindMe systems evaluate this form of conversational recommendation and feedback in a variety of contexts including movie, car, and accommodation recommendation [22]. In the Car Navigator recommender system, for example, individual critiques were also designed to cover multiple features, so that, for instance, a user might request a *sportier* car than the current recommendation, simultaneously constraining features such as *engine size* and *acceleration*. These *compound critiques* obviously allow the

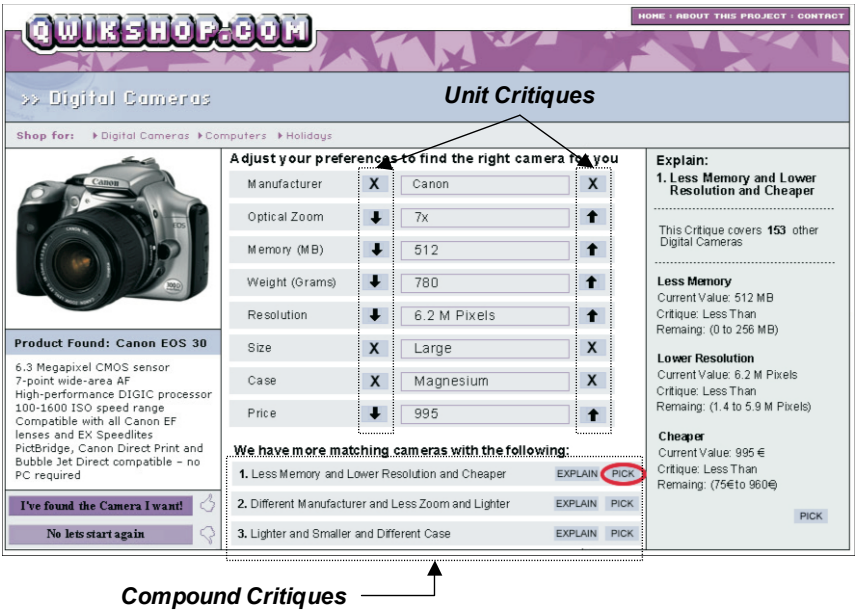

Fig. 11.13. A screenshot of the digital camera recommender system evaluated in [75, 50, 51] which solicits feedback in the form of fixed unit critiques and a set of dynamically generated compound critiques. The former are indicated on either side of the individual camera features, while the latter are presented beneath the main product recommendation as a set of 3 alternatives.

recommender to take larger steps through the product-space, eliminating many more cases than would be possible with a single-feature, *unit critique*, in a single recommendation cycle. Indeed recently the work of [50, 51, 75] has investigated the possibility of automatically generating dynamic compound critiques based on the remaining cases and the user's progress so far; see for example Figure 11.13. In short, this so-called *dynamic critiquing* approach uses data mining techniques to identify groups of unit critiques that reflect common difference patterns between the remaining cases. Evaluation results suggest that using these groups of critiques as compound critiques has the potential to offer significant improvements in recommendation performance by allowing users to navigate more efficiently through complex product-spaces.

Preference-Based Feedback. Perhaps the simplest form of feedback involves the user indicating a simple preference for one recommendation over another. It is also particularly well suited to domains where users have very little domain knowledge, but where they can readily express a preference for what they like when it is recommended. For example, Figure 11.14 shows a set of recommendations from a prototype recommendation service that uses preference-based feedback to help brides-to-be to chose a wedding dress. During each cycle 3 different suggestions are presented along with a set of technical features, and the user can select one recommendation as their preference as a lead into the next cycle. This is a good example of a domain where the average shopper is likely to have limited domain knowledge, at least in terms of the type of technical

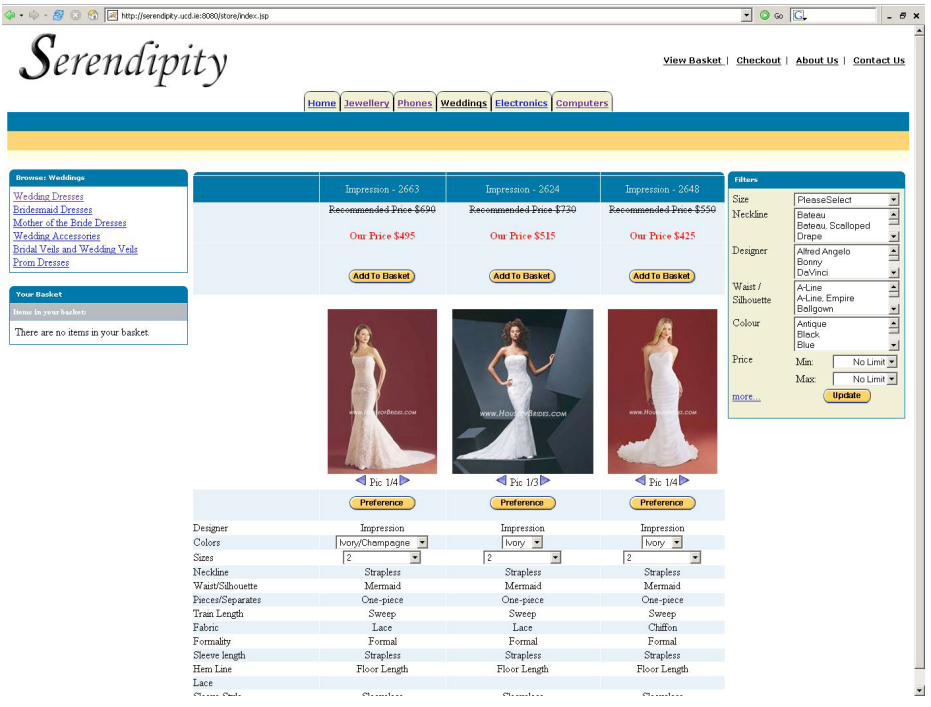

Fig. 11.14. A set of wedding dress suggestions from a prototype recommender system. During each recommendation cycle 3 suggestions are made and the user can indicate which they prefer as a way to initiate the next recommendation cycle.

features that are presented alongside the recommendations. However most brides-to-be will be willing and able to select a preference for one dress over another.

Unfortunately, while this approach carries very little feedback overhead, from a user's perspective, it is ultimately limited in its ability to guide the recommendation process. For example, it will not always be clear why a user has selected one recommendation over another, in terms of the features behind these recommendation. They both may have many features in common and many features that distinguish them. To address this issue the *comparison-based recommendation* work of [52] proposes a variety of query revision strategies that are designed to update the current query as a result of preference-based feedback. For example the most straightforward strategy (*more like this*) simply adopts the preferred case as the new query and proceeds to retrieve the k most similar cases to it for the next cycle. This approach is not very efficient however as it does little to infer the user's true preferences at a feature level. An alternative approach only transfers features from the preferred case if these features are absent from all of the rejected cases, thus allowing the recommender to focus on those aspects of the preferred cases that are unique in the current cycle. Yet another strategy attempts to weight features in the updated query according to how confident the recommender can be that these features are responsible for the user's preference. One particular weighting strategy proposed by [52] depends on the number of alternatives for a given feature

within the current recommendation set. For example, in a digital camera recommender system, if the preferred camera is a *Nikon*, and if there are many alternative manufacturers listed amongst the other $k-1$ recommendations in the current cycle, then the recommender can be more confident that the user is genuinely interested in *Nikons* than if say one or two of the rejected cases were also *Nikons*. Both of these alternative strategies allow for more efficient recommendation sessions than the default *more like this* strategy.

11.4.3 Conversational Management and Mixed-Initiative Systems

Of course while there is a clear distinction between the navigation by asking and navigation by proposing styles of conversation, and between the different forms of feedback that they adopt, this does not limit recommender systems from implementing strategies that combine different conversational styles and methods of feedback. For example, ExpertClerk [86, 87] is designed to mimic the type of interactions that are commonly observed between shoppers and sales clerks in real-life shopping scenarios, and implements a mixture of navigation by asking and navigation by proposing. During the early stages of the recommendation session the user is asked a number of questions in order to identify their broad requirements. Questions are selected using an information theoretic approach based on ID3's information gain heuristic [72, 73]. This is followed by a navigation by proposing stage that includes critiquing-based feedback. Experiments indicate that this combination of navigational styles provides for more efficient recommendation sessions than either approach on its own and, it is argued, constitutes a more natural form of recommendation dialog.

The work of [53, 54, 93] proposes a different departure from the conventional approaches to conversational recommendation. Instead of combining different forms of feedback and conversation style, the *adaptive selection* technique proposes a diversity-enhanced approach to retrieval that has its origins in the work described in [11, 92]; see also Section 11.3 of this chapter. However, rather than simply combining similarity and diversity during each recommendation cycle, diversity is selectively introduced depending on whether or not the current recommendation cycle is judged to be on target with respect to the user's preferences. If the current set of recommendations are not judged by the user to be an improvement on previous recommendations then diversity is increased during the next recommendation cycle in order to provide a broader set of recommendations. The goal is to try to refocus the recommender system on a more satisfactory region of the product-space. If, however, the latest recommendations do appear to be on target then a similarity-based approach to retrieval is used to provide for a more fine-grained exploration of the current region of the product-space. Judging whether or not the current recommendations are on target is achieved using a technique called *preference carrying*. Specifically, the previously preferred product case is carried to the next cycle and if it is reselected then the system assumes that an improvement has not been achieved by the other cases recommended as part of this new cycle. The approach has been shown to have the potential to improve recommendation performance by offering significant reductions in recommendation session lengths when used with either preference-based or critiquing forms of feedback.

In an attempt to further improve the interactive nature of conversational recommender systems researchers have begun to look at how these systems might accommodate a wider variety of *conversational moves* [15], above and beyond the provision of simple forms of user feedback. For example, the Adaptive Place Advisor [37, 98] contemplates a number of different conversational moves in the form of dialog operators. One such operator, *ASK-CONSTRAIN* allows the recommender to ask the user for a particular feature value (standard value elicitation). A different operator (the *QUERY-VALUES* operator) allows the user to ask the recommender for the possible values of a product feature. For example, the recommendation session presented in Figure 11.11 shows examples of the recommender requesting certain feature values and also the user asking for information about the permissible values. Within recommendation sessions the availability of these operators facilitates more flexible interaction models which allow either user or system to seize and cede initiative in a more flexible manner than conventional conversational recommender systems.

This more flexible approach to the development of conversational recommender systems is closely related to recent developments in the area of *mixed-initiative intelligent systems* (see, for example, [2, 97]), which attempt to integrate human and automated reasoning to take advantage of their complementary reasoning styles and different computational strengths. Important issues in mixed-initiative systems include: the division of task responsibility between the human and the intelligent agent; how control and initiative might shift between human and agent, including the support of interrupt-driven or proactive behaviour. Mixed-initiative systems need to maintain a more sophisticated model of the current state of the human and agent(s) involved and require a well-defined set of protocols to facilitate the exchange of information between human and agent(s). For example, the mixed-initiative recommender system, Sermo [15], draws on ideas from conversational analysis techniques in the service of a recommendation framework that combines a conversational recommendation strategy with grammar-based dialog management functions; see also [12]. Sermo's dialog grammar captures the set of legal dialogs between user and recommender system, and the use of a conversational policy facilitates a reasonable balance between over-constrained fixed-role recommendation dialogs and interrupt-driven dialogs that are not sufficiently constrained to deliver coherent recommendation sessions.

11.5 Getting Personal

Personalization technologies promise to offer users a more engaging online experience, one that is tailored to their long-term preferences as well as their short-term needs, both in terms of the information that is presented and the manner in which it is presented. By reading this volume you will gain a detailed understanding of how personalisation techniques have been explored in a wide variety of task contexts, from searching and navigating the Web (see Chapter 6 [61] in this book) to e-commerce (see Chapter 16 [38] in this book), healthcare (see Chapter 15 [23]) and e-learning (Chapter 1 [18] in this book) applications.

Personalization and recommendation technologies are also intimately connected. The ultimate promise of recommendation technology goes beyond the provision of

flexible, conversational information access. Indeed, arguably, the provision of a truly effective recommendation service demands an ability to respond to more than just the short-term needs of the individual. A user's personal preferences should also be considered as an important source of context with which to guide the recommendation process. Ultimately the benefits of personalization in a recommendation context come down to the potential for an improved experience for the user. In theory, personalized recommender systems should be able to respond effectively to users with less feedback; in this sense a user's learned preferences can *"fill in the gaps"*, or at least some of them, during a recommendation dialog, leading to shorter recommendation sessions. For example, recognising that a user is a fan of water sports will help a travel recommender to make more targeted recommendations without soliciting feedback on preferred vacation activities. Moreover, the quality of recommendations should improve because the recommender has access to long-term preference information that may never normally be disclosed during any individual session. For example, as a member of the One World Alliance air-miles programme our holiday-maker may prefer to fly with certain carriers to avail of air-miles. This information may be missing from their query but because it is reflected in their long-term profile these carriers can be promoted during the recommendation process.

Personalized recommender systems are distinguished by their ability to react to the preferences of the individual. Obviously there is a sense in which every recommender system reacts to the preferences of the individual, but does this make it personalized? For example, all of the conversational recommender systems discussed above react to feedback provided by users within each session. As such they react to the personal needs of the user, captured from their direct feedback, and so deliver some degree of personalization in their recommendations. However this is what might be termed *weak personalization* because the absence of a persistent user profile outside of the scope of an individual session precludes the recommender from adapting to preferences that may not be disclosed in that session. Such recommender systems can provide *in-session* personalization only and two users who respond in the same way within a session will receive the same recommendations even if they differ greatly in terms of their long-term preferences. Recommender systems that adopt a model of *strong personalization* must have access to persistent user profiles, which can be used to complement any in-session feedback to guide the recommendation process.

The key then to building a personalized recommender system relies on an ability to learn and maintain a long-term model of a user's recommendation preferences. A wide variety of approaches are available in this regard. For example, user models can represent stereotypical users [24], or they can reflect the preferences of actual users by learning from questionnaires, user ratings or usage traces (e.g., [47, 90]). Moreover, user models can be learned by soliciting long-term preference information from the user directly; for example, by asking the user to weight a variety of areas of interest. Alternatively, models can be learned in a less obtrusive manner by mining their normal online behaviour patterns (e.g., [65, 85]). Further discussion of the user modeling research literature is beyond the scope of this chapter but the interested reader is directed to Chapters 2 [35] and 4 [44] of this book for further material on this important topic.

11.5.1 Case-Based Profiling

Case-based recommendation techniques facilitate a form of *case-based* user profiling that leverages available content descriptions as an intrinsic part of the user profiles. Accordingly a user profile is made up of a set of cases plus some indication of whether the user has liked or disliked these cases. These profiles can then be used in different ways to help influence subsequent recommendations as we will see. First, it is worth highlighting how this approach to profiling stands in contrast to the type of ratings-based profiles that are exploited by collaborative filtering systems (see, for example [5, 30, 47, 74, 77, 85, 89] and also Chapter 9 [83] of this book). Ratings-based profiles are *content-free*, in the sense that they are devoid of any item content; ratings may be associated with a particular item identifier but the information about the item itself (other than the ratings information) is generally not available, at least in pure collaborative filtering systems. In contrast, case-based profiles do contain information about the items themselves because this information is stored within the item cases.

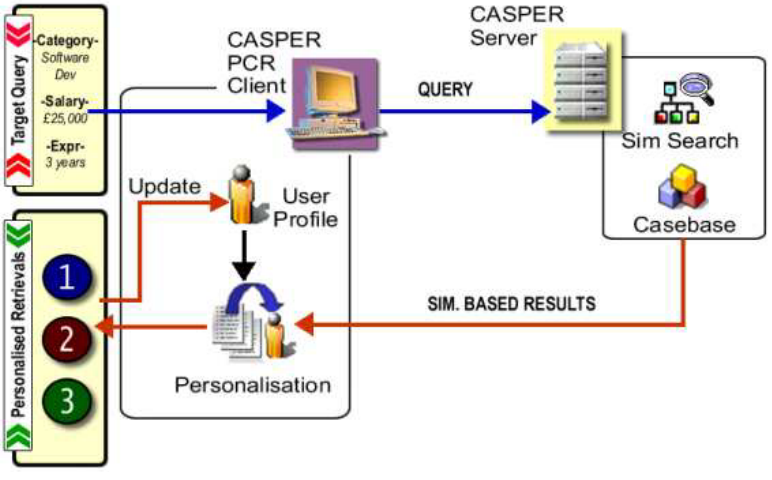

Fig. 11.15. The CASPER case-based recommender system combines case-based recommendation and user profiling in an online recruitment application.

CASPER is online recruitment system (see Figure 11.15) that uses single-shot case-based recommendation to suggest jobs to users based on some initial query. It monitors the responses of users to these recommendations in order to construct case-based profiles that can be used to personalise future recommendation sessions; see [10, 74, 88]. Upon receiving a job recommendation a user has various action choices. For example, they might choose to save the advert, or email it to themselves, or submit their resume online. These actions are interpreted by CASPER's profiling mechanism as examples of positive feedback and stored in a persistent profile as rating-case pairs. Each action is translated into a rating according to how reliable an indicator of user interest it is; for example, submitting a resume as part of an online application is treated as the highest

level of interest. In addition, negative ratings can be provided by rating adverts directly. Thus each user profile is made up of a set of job cases and their ratings for a given user.

CASPER's recommendation process is a two-step one. As indicated in Figure 11.15, CASPER first generates a set of recommendations based on similarity to the user query. Ordinarily these similarity-based results would be suggested to the user. However, the availability of a user profile allows CASPER to further process these recommendations in order to re-rank them according to the preferences encoded in the user's profile. To do this CASPER scores each new recommendation according to how similar it is to the cases stored in the user's profile (see Figure 11.16). This score measures the relevance of a new recommendation to the user in question. Recommendations that are similar to cases that the user has liked are preferred over recommendations that are similar to cases that the user has disliked, for example. Thus, if, over time the user has responded positively to jobs in a particular region, or with a particular salary range then, in the future, recommendations that include similar features will tend to be prioritised even if these features are absent from the current query.

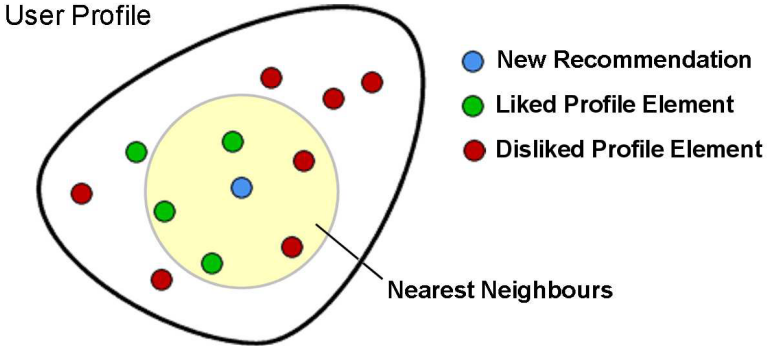

Fig. 11.16. To judge the relevance of a new recommendation, which has been suggested because of its similarity to the target query, CASPER compares the recommendation to the other cases that are similar in the user's profile and score the recommendation according to the relative distribution of positive and negative ratings within this set of similar cases.

The final ordering of recommendations can then be influenced by a combination of their similarity to the user's target query and their relevance to the user's past preferences. Live and artificial-user studies have demonstrated that this form of personalization has the potential to significantly improve the quality of CASPER's recommendations. As an aside, it is worth highlighting that in CASPER profiles are stored and updated on the client-side and the personalized re-ranking is performed as a client-side process. This enhances the system's privacy characteristics because personal preferences do not have to be revealed to the recommendation server.

A similar form of case-based profiling is used in the Personal Travel Assistant (PTA) [27, 28]. Like CASPER, the Personal Travel Assistant uses its profiles to re-order an initial set of recommendations based on the preferences that are encoded within a profile. A complimentary approach to recommendation reordering is proposed by [79, 80],

but instead of relying on the target user's profile, reordering is performed with reference to a set of similar recommendation sessions that have occurred in the past with similar users.

11.5.2 Profiling Feature Preferences

The methods discussed in the previous section refer to the uses of cases themselves as part of the user profile; the user profile is made up of a set of positive (and possibly negative) cases. Collectively these rated cases capture a user's preferences. An alternative, and somewhat more direct approach to profiling in case-based recommender systems, sees the use of user profiles that attempt to capture feature level preferences. For example, the user models employed in the Adaptive Place Advisor [37, 98] capture the preferences that a user may have in relation to items, attributes and their values including the relative importance of a particular attribute, the preferred values for a particular attribute, and specific preferred items encoded as frequency distributions. For instance, for a restaurant, the user model might indicate that the type of *cuisine* has an importance weight of 0.4, compared to say the *parking* facilities, which might have a preference weight of only 0.1. Moreover a user may have a 0.35 preference weight for Italian cuisine but only a 0.1 weighting for German food. Individual item preferences are represented as the proportion of times that an item has been accepted in past recommendations. These preferences then guide the recommendation engine during item selection. Item, attribute and value preferences are all expressed as part of an item selection similarity metric so that items which have been preferred in the past, or that have important attributes and/or preferred values for these attributes, are prioritised over items that have weaker preferences.

When it comes to updating the user model, the Adaptive Place Advisor interprets various user actions as preferences for or against certain items, attributes or values. For example, when a user accepts a recommendation, this is interpreted as a preference, not just for this item, but also for its particular attributes and values. However, when a user rejects an item, this is interpreted as a rejection of the item itself, rather than a rejection of its attribute-value combinations. Early evaluations indicate that the combination of Adaptive Place Advisor's conversational recommendation technique and its personalization features has the potential to improve recommendation performance, with efficiency improvements noted as subjects learned how to interact with the system and as their user models were updated as a result of these interactions.

11.6 Conclusions

Case-based recommendation is a form of content-based recommendation that emphasises the use of structured representations and similarity-based retrieval during recommendation. In this chapter we have attempted to cover a number of important lines of research in case-based recommendation, from traditional models of single-shot, similarity-based recommendation to more sophisticated conversational recommendation models and personalization techniques. This research has helped to distinguish

case-based recommendation techniques from their content-based and collaborative filtering cousins.

Of course it has only been possible to scratch the surface of this vibrant area of research and many emerging topics have been omitted. For example, there is now an appreciation that no one recommendation technology or strategy is likely to be optimal in any given recommendation scenario. As a result considerable attention has been paid to the prospect of developing hybrid recommendation strategies that combine individual approaches, such as content-based and collaborative filtering techniques; see for example Chapter 12 [20] in this book.

Another area of recent focus concerns the ability of case-based recommender systems (and indeed recommender systems in general) to *explain* the results of their reasoning. Certainly, when it comes to buying expensive goods, people expect to be skillfully steered through the options by well-informed sales assistants that are capable of balancing the user's many and varied requirements. But in addition users often need to be educated about the product space, especially if they are to come to understand what is available and why certain options are being recommended by the sales-assistant. Thus recommender systems also need to educate users about the product space: to *justify* their recommendations and *explain* the reasoning behind their suggestions; see, for example, [29, 36, 57, 59, 60, 76, 82, 95]

In summary then, case-based recommendation provides for a powerful and effective form of recommendation that is well suited to many product recommendation scenarios. As a style of recommendation, its use of case knowledge and product similarity, makes particular sense in the context of interactive recommendation scenarios where recommender system and user must collaborative in a flexible and transparent manner. Moreover, the case-based approach enjoys a level of transparency and flexibility that is not always possible with other forms of recommendation.

Acknowledgements. This work was supported by Science Foundation Ireland under Grant No. 03/IN.3/I361.

References

1. Aamodt, A., Plaza, E.: Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AI Communications **7(1)** (1994) 39–52
2. Aha, D.W.: Proceedings of the Workshop in Mixed-Initiative Case-Based Reasoning, Workshop Programme at the 6th European Conference in Case-Based Reasoning. http://home.earthlink.net/ dwaha/research/meetings/eccbr02-micbrw/ (2002)
3. Aha, D.W., Maney, T., Breslow, L.A.: Supporting Dialogue Inferencing in Conversational Case-Based Reasoning. In Smyth, B., Cunningham, P., eds.: Proceedings of the 4th European Workshop on Case-Based Reasoning, Springer-Verlag (1998) 262–273
4. Aha, D., Breslow, L.: Refining Conversational Case Libraries. In Leake, D., Plaza, E., eds.: Proceedings of the 2nd International Conference on Case-Based Reasoning, Springer-Verlag (1997) 267–278
5. Balabanovic, M., Shoham, Y.: FAB: Content-Based Collaborative Recommender. Communications of the ACM **40(3)** (1997) 66–72

6. Bergmann, R., Cunningham, P.: Acquiring customer's requirements in electronic commerce. Artificial Intelligence Review **18**(3–4) (2002) 163–193
7. Bergmann, R., Richter, M., Schmitt, S., Stahl, A., Vollrath, I.: Utility-Oriented Matching: A New Research Direction for Case-Based Reasoning. In: Proceedings of the German Workshop on Case-Based Reasoning. (2001) 264–274
8. Billsus, D., Pazzani, M.J.: Adaptive news access. In Brusilovsky, P., Kobsa, A., Nejdl, W., eds.: The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag (2007) this volume
9. Billsus, D., Pazzani, M.J., Chen, J.: A learning agent for wireless news access. In: IUI '00: Proceedings of the 5th international conference on Intelligent user interfaces, ACM Press (2000) 33–36
10. Bradley, K., Rafter, R., Smyth, B.: Case-Based User Profiling for Content Personalization. In Brusilovsky, P., Stock, O., Strapparava, C., eds.: Proceedings of the 1st International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH2000), Springer-Verlag (2000) 62–72
11. Bradley, K., Smyth, B.: Improving Recommendation Diversity. In O'Donoghue, D., ed.: Proceedings of the 12th National Conference in Artificial Intelligence and Cognitive Science, Maynooth, Ireland (2001) 75–84
12. Branting, K., Lester, J., Mott, B.: Dialog Management for Conversational Case-Based Reasoning. In Funk, P., Calero, P.A.G., eds.: Proceedings of the 7th European Conference on Case-Based Reasoning, Springer-Verlag (2004) 77–90
13. Branting, K.: Acquiring customer preferences from return-set selections. In Aha, D.W., Watson, I., eds.: Proceedings of the 4th International Conference on Case-Based Reasoning, Springer-Verlag (2001) 59–73
14. Bridge, D.: Diverse Product Recommendations Using an Expressive Language for Case-Retrieval. In Craw, S., Preece, A., eds.: Proceedings of the 6th European Conference on Case-Based Reasoning, Springer-Verlag (2002) 43–57
15. Bridge, D.: Towards conversational recommender systems: A dialogue grammar approach. In D.W.Aha, ed.: Proceedings of the Workshop in Mixed-Initiative Case-Based Reasoning, Workshop Programme at the 6th European Conference in Case-Based Reasoning. (2002) 9–22
16. Bridge, D., Goker, M., McGinty, L., Smyth, B.: Case-based recommender systems. Knowledge Engineering Review **20**(3) (2006) 315–320
17. Bridge, D.: Product recommendation systems: A new direction. In R.Weber, Wangenheim, C., eds.: Procs. of the Workshop Programme at the Fourth International Conference on Case-Based Reasoning. (2001) 79–86
18. Brusilovsky, P., Millan, E.: User models for adaptive hypermedia and adaptive educational systems. In Brusilovsky, P., Kobsa, A., Nejdl, W., eds.: The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag (2007) this volume
19. Burke, R.: Conceptual Indexing and Active Retrieval of Video for Interactive Learning Environments. Knowledge-Based Systems **9(8)** (1996) 491–499
20. Burke, R.: Hybrid web recommender systems. In Brusilovsky, P., Kobsa, A., Nejdl, W., eds.: The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag (2007) this volume
21. Burke, R., Hammond, K., Young, B.: Knowledge-based navigation of complex information spaces. In: Proceedings of the Thirteenth National Conference on Artificial Intelligence, AAAI Press/MIT Press (1996) 462–468 Portland, OR.
22. Burke, R., Hammond, K., Young, B.: The FindMe Approach to Assisted Browsing. Journal of IEEE Expert **12(4)** (1997) 32–40

23. Cawsey, A., Grasso, F., Paris, C.: Adaptive information for consumers of healthcare. In Brusilovsky, P., Kobsa, A., Nejdl, W., eds.: The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag (2007) this volume
24. Chin, D.N.: Acquiring user models. Artificial Intelligence Review **7**(3-4) (1989) 185–197
25. Cohen, W.W., Schapire, R.E., Singer, Y.: Learning to order things. In: NIPS '97: Proceedings of the 1997 conference on Advances in neural information processing systems 10, MIT Press (1998) 451–457
26. Cotter, P., Smyth, B.: Waping the web: Content personalisation for wap-enabled devices. In: Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, Trento, Italy., Springer-Verlag (2000) 98–108
27. Coyle, L., Cunningham, P.: Improving Recommendation Ranking by Learning Personal Feature Weights. In Funk, P., Calero, P.A.G., eds.: Proceedings of the 7th European Conference on Case-Based Reasoning, Springer-Verlag (2004) 560–572
28. Coyle, L., Cunningham, P., Hayes, C.: A Case-Based Personal Travel Assistant for Elaborating User Requirements and Assessing Offers. In Craw, S., Preece, A., eds.: Proceedings of the 6th European Conference on Case-Based Reasoning, Springer-Verlag (2002) 505–518
29. Cunningham, P., Doyle, D., Loughrey, J.: An Evaluation of the Usefulness of Case-Based Explanation. . In Ashley, K., Bridge, D., eds.: Case-Based Reasoning Research and Development. LNAI, Vol. 2689., Springer-Verlag (2003) 191–199
30. Dahlen, B., Konstan, J., Herlocker, J., Good, N., Borchers, A., Riedl, J.: Jump-starting movieLens: User benefits of starting a collaborative filtering system with "dead-data". In: , University of Minnesota TR 98-017 (1998)
31. de Mantaras, R.L., McSherry, D., Bridge, D., Leake, D., Smyth, B., Craw, S., Faltings, B., Maher, M.L., Cox, M.T., Forbus, K., Keane, M., Aamody, A., Watson, I.: Retrieval, reuse, revision, and retention in case-based reasoning. Knowledge Engineering Review **20**(3) (2006) 215–240
32. Doyle, M., Cunningham, P.: A dynamic approach to reducing dialog in on-line decision guides. In Blanzieri, E., Portinale, L., eds.: Proceedings of the 5th European Workshop on Case-Based Reasoning, Springer-Verlag (2000) 49–60
33. Faltings, B., Pu, P., Torrens, M., Viappiani, P.: Design Example-Critiquing Interaction. In: Proceedings of the International Conference on Intelligent User Interface(IUI-2004), ACM Press (2004) 22–29 Funchal, Madeira, Portugal.
34. Fox, S., Leake, D.B.: Using Introspective Reasoning to Refine Indexing. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, Morgan Kaufmann (1995) 391 – 397
35. Gauch, S., Speretta, M., Chandramouli, A., Micarelli, A.: User profiles for personalized information access. In Brusilovsky, P., Kobsa, A., Nejdl, W., eds.: The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag (2007) this volume
36. Gervas, P., Gupta, K.: Proceedings European Conference on Case-Based Reasoning (ECCBR-04) Explanation Workshop. (2004) Madrid, Spain.
37. Goker, M., Thompson, C.A.: Personalized Conversational Case-Based Recommendation. In Blanzieri, E., Portinale, L., eds.: Proceedings of the 5th European Workshop on Case-Based Reasoning, Springer-Verlag (2000) 99–111
38. Goy, A., Ardissono, L., Petrone, G.: Personalization in e-commerce applications. In Brusilovsky, P., Kobsa, A., Nejdl, W., eds.: The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag (2007) this volume

39. Hinkle, D., Toomey, C.: Applying case-based reasoning to manufacturing. Artificial Intelligence Magazine **16**(1) (1995) 65–73
40. Hurley, G., Wilson, D.C.: DubLet: An online CBR system for rental property accommodation. In Aha, D.W., Watson, I., eds.: Proceedings of the 4th International Conference on Case-Based Reasoning, Springer-Verlag (2001) 660–674
41. Jameson, A., Smyth, B.: Recommending to groups. In Brusilovsky, P., Kobsa, A., Nejdl, W., eds.: The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag (2007) this volume
42. Juell, P., Paulson, P.: Using reinforcement learning for similarity assessment in case-based systems. IEEE Intelligent Systems **18**(4) (2003) 60–67
43. Kim, Y., Ok, S., Woo, Y.: A Case-Based Recommender using Implicit Rating Techniques. In DeBra, P., Brusilovsky, P., Conejo, R., eds.: Proceedings of the 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH2002) , Springer-Verlag (2002) 62–72
44. Kobsa, A.: Generic user modeling systems. In Brusilovsky, P., Kobsa, A., Nejdl, W., eds.: The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag (2007) this volume
45. Kolodner, J.: Judging which is the "best" case for a case-based reasoner. In: Proceedings of the Second Workshop on Case-Based Reasoning, Morgan Kaufmann (1989) 77–81
46. Kolodner, J.: Case-Based Reasoning. Morgan Kaufmann (1993)
47. Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gorgan, L., Riedl, J.: GroupLens: Applying collaborative filtering to Usenet news. Communications of the ACM **40(3)** (1997) 77–87
48. Leake, D.: Case-Based Reasoning: Experiences,Lessons and Future Directions. AAAI/MIT Press (1996)
49. Leake, D.B.: Constructive Similarity Assessment: Using Stored Cases to Define New Situations. In: Proceedings of the 14th Annual Conference of the Cognitive Science Society, Lawrence Earlbaum Associates (1992) 313–318
50. McCarthy, K., Reilly, J., McGinty, L., Smyth, B.: On the Dynamic Generation of Compound Critiques in Conversational Recommender Systems. In DeBra, P., ed.: Proceedings of the 3rd International Conference on Adaptive Hypermedia and Web-Based Systems (AH04), Springer-Verlag (2004) 176–184 Eindhoven, The Netherlands.
51. McCarthy, K., Reilly, J., McGinty, L., Smyth, B.: Experiments in dynamic critiquing. In: IUI '05: Proceedings of the 10th international conference on Intelligent user interfaces, ACM Press (2005) 175–182
52. McGinty, L., Smyth, B.: Comparison-Based Recommendation. In Craw, S., Preece, A., eds.: Proceedings of the 6th European Conference on Case-Based Reasoning, Springer-Verlag (2002) 575–589
53. McGinty, L., Smyth, B.: On The Role of Diversity in Conversational Recommender Systems. In Ashley, K.D., Bridge, D.G., eds.: Proceedings of the 5th International Conference on Case-Based Reasoning, Springer-Verlag (2003) 276–290
54. McGinty, L., Smyth, B.: Tweaking Critiquing. In: Proceedings of the Workshop on Personalization and Web Techniques at the International Joint Conference on Artificial Intelligence (IJCAI-03), Morgan-Kaufmann (2003) Acapulco, Mexico.
55. McSherry, D.: Increasing Recommendation Diversity Without Loss of Similarity. In: Proceedings of the Sixth UK Workshop on Case-Based Reasoning. (2001) 23–31 Cambridge, UK.
56. McSherry, D.: Diversity-Conscious Retrieval. In Craw, S., Preece, A., eds.: Proceedings of the 6th European Conference on Case-Based Reasoning, Springer-Verlag (2002) 219–233
57. McSherry, D.: Explanation in Case-Based Reasoning: An Evidential Approach. . In Lees, B., ed.: Proceedings of the 8th UK Workshop on Case-Based Reasoning. (2003) Cambridge, UK.

58. McSherry, D.: Similarity and Compromise. In Ashley, K.D., Bridge, D.G., eds.: Proceedings of the 5th International Conference on Case-Based Reasoning, Springer-Verlag (2003) 291–305
59. McSherry, D.: Explaining the Pros and Cons of Conclusions in CBR. . In Calero, P.A.G., Funk, P., eds.: Proceedings of the 6th European Conference on Case-Based Reasoning (ECCBR-04), Springer-Verlag (2004) 317–330 Madrid, Spain.
60. McSherry, D.: Incremental Relaxation of Unsuccessful Queries. . In Calero, P.A.G., Funk, P., eds.: Proceedings of the 6th European Conference on Case-Based Reasoning (ECCBR-04), Springer-Verlag (2004) 331–345 Madrid, Spain.
61. Micarelli, A., Gasparetti, F., Sciarrone, F., gauch, S.: Personalized search on the world-wide web. In Brusilovsky, P., Kobsa, A., Nejdl, W., eds.: The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag (2007) this volume
62. Mobasher, B.: Data mining for web personalization. In Brusilovsky, P., Kobsa, A., Nejdl, W., eds.: The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag (2007) this volume
63. Mooney, R.J., Roy, L.: Content-based book recommending using learning for text categorization. In: DL '00: Proceedings of the fifth ACM conference on Digital libraries, ACM Press (2000) 195–204
64. Munoz-Avila, H., Aha, D., Breslow, L.: Integrating Conversational Case Retrieval with Generative Planning. In Blanzieri, E., Portinale, L., eds.: Proceedings of the 5th European Workshop on Case-based Reasoning, Springer-Verlag (2000) 210–221
65. Nichols, D.: Implicit rating and filtering. In: Proceedings of 5th DELOS Workshop on Filtering and Collaborative Filtering. (November 1997)
66. O'Sullivan, D., Smyth, B., Wilson, D.: In-Depth Analysis of Similarity Knowledge and Metric Contributions to Recommender Performance. In: Proceedings of the 17th International FLAIRS Conference, May 17-19, Miami Beach, Florida, FLAIRS 2004 CD, AAAI Press. (2004)
67. O'Sullivan, D., Wilson, D., Smyth, B.: Improving Case-Based Recommendation: A Collaborative Filtering Approach. In Craw, S., Preece, A.D., eds.: 6th European Conference on Case-Based Reasoning. Volume 2416., Springer-Verlag (2002) 278 – 291
68. O'Sullivan, D., Wilson, D., Smyth, B.: Preserving Recommender Accuracy and Diversity in Sparse Datasets. In Russell, I., Haller, S., eds.: Proceedings of the 16th International FLAIRS Conference, AAAI Press (2003) 139 – 144
69. Pazzani, M.J., Billsus, D.: Content-based recommendation systems. In Brusilovsky, P., Kobsa, A., Nejdl, W., eds.: The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag (2007) this volume
70. Pu, P., Faltings, B.: Decision Tradeoff Using Example Critiquing and Constraint Programming. Special Issue on User-Interaction in Constraint Satisfaction. CONSTRAINTS: an International Journal. **9(4)** (2004)
71. Quinlan, J.R.: Induction of decision trees. Machine Learning **1** (1986) 81–106
72. Quinlan, J.R.: Learning decision tree classifiers. ACM Comput. Surv. **28**(1) (1996) 71–72
73. Quinlan, J.R.: C4.5: programs for machine learning. Morgan Kaufmann Publishers Inc. (1993)
74. Rafter, R., Bradley, K., Smyth, B.: Automatic Collaborative Filtering Applications for Online Recruitment Services. In P. Brusilovsky, O.S., Strapparava, C., eds.: Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-based Systems, Springer-Verlag (2000) 363–368
75. Reilly, J., McCarthy, K., McGinty, L., Smyth, B.: Dynamic Critiquing. In Funk, P., Calero, P.A.G., eds.: Proceedings of the 7th European Conference on Case-Based Reasoning, Springer-Verlag (2004) 763–777

76. Reilly, J., McCarthy, K., McGinty, L., Smyth, B.: Explaining compound critiques. Artificial Intelligence Review (In Press)
77. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: GroupLens: An open architecture for collaborative filtering of netnews. In: Proceedings of the 1994 Conference on Computer Supported Collaborative Work. (1994) 175–186
78. Resnick, P., Varian, H.R.: Recommender systems. CACM **40**(3) (1997) 56–58
79. Ricci, F., Venturini, A., Cavada, D., Mirzadeh, N., Blaas, D., Nones, M.: Product Recommendation with Interactive Query Management and Twofold Similarity. In Ashley, K.D., Bridge, D.G., eds.: Proceedings of the 5th International Conference on Case-Based Reasoning, Springer-Verlag (2003) 479–493
80. Ricci, F., Arslan, B., Mirzadeh, N., Venturini, A.: Itr: A case-based travel advisory system. In Craw, S., Preece, A., eds.: Proceedings of the 6th European Conference on Case-Based Reasoning, Springer-Verlag (2002) 613–627
81. Ricci, F., Avesani, P.: Learning a local similarity metric for case-based reasoning. In: ICCBR '95: Proceedings of the First International Conference on Case-Based Reasoning Research and Development, Springer-Verlag (1995) 301–312
82. Roth-Berghofer, T.R.: Explanations and Case-Based Reasoning: Foundational Issues. In Funk, P., Calero, P.A.G., eds.: Proceedings of the 7th European Conference on Case-Based Reasoning, Springer-Verlag (2004) 389–403
83. Schafer, J.B., Frankowski, D., Herlocker, J., Sen, S.: Collaborative filtering recommender systems. In Brusilovsky, P., Kobsa, A., Nejdl, W., eds.: The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag (2007) this volume
84. Schmitt, S.: *simVar*; a similarity-influenced question selection criterion for e-sales dialogs. Artificial Intelligence Review **18**(3–4) (2002) 195–221
85. Shardanand, U., Maes, P.: Social Information Filtering: Algorithms for Automating "Word of Mouth". In: Proceedings of the Denver ACM CHI 1995. (1995) 210–217
86. Shimazu, H.: ExpertClerk : Navigating Shoppers' Buying Process with the Combination of Asking and Proposing. In Nebel, B., ed.: Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01), Morgan Kaufmann (2001) 1443–1448 Seattle, Washington, USA.
87. Shimazu, H., Shibata, A., Nihei, K.: ExpertGuide: A conversational case-based reasoning tool for developing mentors in knowledge spaces. Applied Intelligence **14(1)** (2002) 33–48
88. Smyth, B., Bradley, K., Rafter, R.: Personalization techniques for online recruitment services. Commun. ACM **45**(5) (2002) 39–40
89. Smyth, B., Cotter, P.: Surfing the Digital Wave: Generating Personalized Television Guides Using Collaborative, Case-based Recommendation. In: Proceedings of the Third International Conference on Case-based Reasoning. (1999)
90. Smyth, B., Cotter, P.: A Personalized TV Listings Service for the Digital TV Age. Journal of Knowledge-Based Systems **13(2-3)** (2000) 53–59
91. Smyth, B., Keane, M.: Adaptation-Guided Retrieval: Questioning the Similarity Assumption in Reasoning. Artificial Intelligence **102** (1998) 249–293
92. Smyth, B., McClave, P.: Similarity v's Diversity. In Aha, D., Watson, I., eds.: Proceedings of the 3rd International Conference on Case-Based Reasoning, Springer-Verlag (2001) 347–361
93. Smyth, B., McGinty, L.: The Power of Suggestion. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-03), Morgan-Kaufmann (2003) 127–132 Acapulco, Mexico.
94. Smyth, B., Cotter, P.: A personalized television listings service. Communications of the ACM **43**(8) (2000) 107–111

95. Sørmo, F., Cassens, J.: Explanation goals in case-based reasoning. In Gervás, P., Gupta, K.M., eds.: Proceedings of the ECCBR 2004 Workshops, Departamento de Sistemas Informáticos y Programación, Universidad Complutense Madrid (2004) 165–174
96. Stahl, A.: Learning feature weights from case order feedback. In Aha, D.W., Watson, I., eds.: Proceedings of the 4th International Conference on Case-Based Reasoning. (2001) 502–516
97. Tecuci, G., Aha, D.W., Boicu, M., Cox, M., Ferguson, G., Tate, A.: Proceedings of the Workshop on Mixed-Initiative Intelligent Systems, Workshop Programme at the 18th International Joint Conference on Artificial Intelligence. http://lalab.gmu.edu/miis/ (2003)
98. Thompson, C.A., Goker, M.H., Langley, P.: A Personalized System for Conversational recommendation. Journal of Artificial Intelligence Research **21** (2004) 1–36
99. Veloso, M., Munoz-Avila, H., Bergmann, R.: Case-Based Planning: Methods and Systems. AI Communications **9(3)** (1996) 128–137
100. Vollrath, I., Wilke, W., Bergmann, R.: Case-based reasoning support for online catalog sales. IEEE Internet Computing **2**(4) (1998) 45–54
101. Watson, I., ed.: Applying Case-Based Reasoning: Techniques for Enterprise Systems. Morgan Kaufmann (1997)
102. Wettschereck, D., Aha, D.W.: Weighting features. In Veloso, M.M., Aamodt, A., eds.: Proceedings of the 1st International Conference on Case-Based Reasoning, Springer-Verlag (1995) 347–358
103. Wilke, W., Lenz, M., Wess, S.: Intelligent sales support with CBR. In et al., M.L., ed.: Case-Based Reasoning Technology: From Foundations to Applications. Springer-Verlag (1998) 91–113