

# Experience-Based Critiquing: Reusing Critiquing Experiences to Improve Conversational Recommendation

Kevin McCarthy, Yasser Salem, and Barry Smyth

CLARITY: Centre for Sensor Web Technologies  
School of Computer Science and Informatics  
University College Dublin, Belfield, Dublin 4, Ireland  
{kevin.mccarthy,yasser.salem,barry.smyth}@ucd.ie  
<http://www.clarity-centre.org>

**Abstract.** Product recommendation systems are now a key part of many e-commerce services and have proven to be a successful way to help users navigate complex product spaces. In this paper, we focus on critiquing-based recommenders, which permit users to *tweak* the features of recommended products in order to refine their needs and preferences. In this paper, we describe a novel approach to reusing past critiquing histories in order to improve overall recommendation efficiency.

## 1 Introduction

Today, e-commerce services rely on recommender systems to help users to navigate complex product spaces. Amazon's use of recommendation technologies is well documented [5] and the recent Netflix competition [1] highlights the value of sophisticated recommendation techniques in the commercial world. The recommender systems community has explored a diverse space of different recommendation techniques, from *collaborative filtering* methods, which rely on simple ratings-based profiles to generate recommendations from similar users, to *content-based* methods, which rely on the availability of product knowledge, generally in the form of detailed descriptions, to make recommendations.

Most deployed recommender systems are *single-shot*, in the sense that the job of the recommender is to produce a single list of recommendations for the user [4,14,15]. The single-shot strategy is well-adapted to the recommendation of simple products and services, such as ringtones, movies, books, etc., but it is not so well suited to recommending more complex items. In more complex recommendation scenarios it is appropriate to offer the user an opportunity to provide feedback, to refine their needs and preferences, based on recent recommendations. In response researchers have developed so-called *conversational recommendation* strategies [2,8,9,16] to support this type of recommendation scenario. Briefly, users participate in a *recommendation dialogue*, receiving recommendations and providing feedback in order to inform a new set of recommendations. Different approaches to conversational recommendation can be distinguished by their use

of different types of feedback and one type of feedback that forms the basis of this work is *critiquing*. Critiquing in a simple form of feedback which facilitates a “*show me more like this but ...*” type of response. Recently there has been renewed interest in critiquing [3,7,10,11,13,19] because of its many advantages: it is simple to implement for a variety of interface types, and it can be appropriate for users who are not experts in a given product domain. However, as we shall see, traditional approaches to critiquing can lead to protracted dialog sessions leading researchers to seek out ways of improving the performance of critiquing-based recommender systems [7,19].

In this paper, we are interested in improving the efficiency of critiquing-based recommender systems without introducing additional critiquing complexity. Our starting point is the idea that the critiquing histories, or experiences, of users carry important information about feature preferences and trade-offs and we consider how these experiences can be usefully reused to bias the recommendation process. In the following sections, we describe one such technique for harnessing critiquing experiences during recommendation and demonstrate its effectiveness, relative to conventional critiquing, on a real-world restaurant dataset.

## 2 Related Work

Recommender systems are a common way to promote products or services that may be of interest to a user, usually based on some profile of interests. The single-shot approach, which produces a ranked list of recommendations, is limited by design. It works well when a user’s needs are clear, but it is less suitable when a user’s needs are not well known, or where they are likely to evolve during the course of a session. In these scenarios it is more appropriate to engage the user in a recommendation dialog so that incremental feedback can be used to refine recommendations. This type of conversational recommender system is much better suited to help users navigate more complex product spaces.

Various forms of feedback are used in conversational recommender systems: value elicitation, ratings-based feedback, preference-based feedback and critiquing [17]. Systems which employ value elicitation expect users to input specific values for product features, e.g. *hard-drive = 320GB*. This is a rich form of feedback so the user must possess a high level of domain knowledge to use it effectively. In contrast, ratings-based feedback is a much simpler form of feedback, preferred by most collaborative filtering systems. Users assign a simple rating, e.g. *3 stars out of 5*, to indicate their satisfaction with the recommendation; see, for example, [4]. With ratings-based feedback, the user does not require detailed domain knowledge, since they are not commenting on specific features. Instead they simply provide an overall recommendation rating. Preference-based feedback is a special case of this in which, instead of rating a set of recommendations, the user simply indicates their preferred recommendation [8]. It is a low cost form of recommendation that requires minimal domain knowledge, just an ability to distinguish good from bad recommendations. However, it is clearly limited in its information content, as it is not always apparent why the user has selected one recommendation over others. In this paper, we look at an altogether different form of feedback; critiquing,

which strikes a balance between ease of feedback and the information content of the feedback. Simply put, critiquing allows users to indicate a *directional preference* with respect to a particular product feature. For example, in a restaurant recommender a user might respond to a given recommendation by asking for a new suggestion that is *cheaper*. In this case the user is critiquing the *price* feature, asking for a new restaurant with a lower price; this is the standard form of critiquing, which is also called *unit critiquing* because the user critiques a single feature at a time. The critique acts as a filter over the remaining recommendable items. The next recommendation will be compatible with the critique while being maximally similar to the previous recommendation.

The screenshot displays the QwikShop.com interface for digital cameras. It features a navigation bar with 'HOME', 'ABOUT THIS PROJECT', and 'CONTACT'. Below the site name, there are links for 'Shop for: Digital Cameras, Holidays, PCs'. The main content area is divided into several sections:

- Product Image:** A Canon EOS-300D camera.
- Item Found: CASE2**
- Specifications:**
  - 6.3 Megapixel CMOS sensor
  - 7-point wide-area AF
  - High-performance DIGIC processor
  - 100-1600 ISO speed range
  - Compatible with all Canon EF lenses and EX Speedlites
  - PictBridge, Canon Direct Print and Bubble Jet Direct compatible - no PC required
- Adjust your preferences in product for you! (Unit Critiques):**
  - Manufacturer: Canon
  - Model: EOS-300D
  - Price (\$): 871.0
  - Format: SLR
  - Resolution (M Pixels): 6.29
  - Optical Zoom (X): 10.0
  - Digital Zoom (X): 0.0
  - Weight (grams): 645.0
  - Storage Type: Compact Flash
  - Storage Included (MB): 0.0
- Compound Critiques:**
  - 1. Less Optical Zoom & More Digital Zoom & A Different Storage Type (139)** [PRECH] [EXPLAIN]
  - 2. A Lower Resolution & A Different Format & Cheaper (169)** [PRECH] [EXPLAIN]
  - 3. A Different Manufacturer & Less Optical Zoom & More Storage (167)** [PRECH] [EXPLAIN]

Fig. 1. Critique-based recommender system specialising in digital cameras

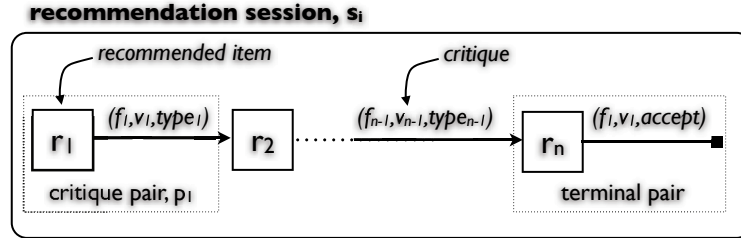
Critiquing has been evaluated in many e-commerce situations, from choosing a restaurant to purchasing a camera [3,7,10]. For example, Figure 1 shows a screen-shot of a typical critique-based recommender system for recommending digital cameras. The current recommendation is displayed in the main window, described by a range of features (i.e. manufacturer, resolution, price, etc.), and the critiques are presented on either side of the feature values (marked “Unit Critiques”). The user can choose one of these critiques to refine their query (e.g. More than 6.29M Pixels). The recommender system uses the critique as a constraint over the value-space of the feature when choosing the next recommendation. A key advantage is that the user does not need to provide a specific value for a product feature, so it demands a lower level of product knowledge

and domain expertise. Critiquing also has a simple interaction mechanism which can be accommodated in a variety of interface styles. However, while critiques help the recommender to narrow its search, progress through a complex product space can be slow, leading to protracted sessions and unhappy users [11]. Moreover, users often change their mind within a session, which can limit the effectiveness of certain approaches to critiquing; see [12].

Recently a number of researchers have started to look at how to make critiquing more efficient by allowing users to provide feedback on multiple features with a single critique. For instance, the work of [7,11,13,19] describes the generation of so-called *compound critiques* from collections of individual *unit critiques*. The screenshot of the recommender system in Figure 1 shows compound critiques which allow the user to cover multiple feature critiques at the same time (e.g. “*A Lower Resolution & A Different Format & Cheaper*”). Not only do compound critiques allow the user to make larger jumps through the product space, they also help to clarify the different trade-offs that exist between features, which can help to improve a user’s domain knowledge. Compound critiques are dynamically generated during each recommendation cycle. For example [7,11] use association rule mining in order to discover compound critiques whereas [19] use a version of multi-attribute utility theory (MAUT). Live user studies of compound critiquing indicate that when users utilise compound critiques (in 35% - 55% of cycles) they go on to benefit from much shorter sessions (20%-50% reductions) compared to users who ignored compound critiques [6]. While such benefits speak to the potential value of compound critiquing, in practice these benefits are limited to those users who avail of compound critiques and many users do not. Hence, in this paper we will describe an alternative solution to this problem, one that does not rely on new feedback options, but rather focuses on improving the traditional unit critiquing. We introduce a new source of knowledge to the critiquing process, namely the critiquing experiences of other users, on the assumption that these experiences may encode meaningful patterns of critiques which may help us to short-cut the standard critiquing process.

### 3 Experience-Based Critiquing

Inspired by ideas in case-based reasoning our proposed experience-based critiquing technique attempts to harness a new source of knowledge during the critiquing process, namely the critiquing experiences of other users. We will focus on past critiquing sessions that have been successful – in the sense that they have led to a purchase decision, for example, and our basic assumption is that these successful experiences must encode useful patterns of critiques, which may help us to short-cut the standard critiquing process for future users. This strand of research borrows from work on mining web logs for request sequences, to use for predicting web pages for caching and prefetching [18]. In what follows we will describe a novel technique to leverage these experiences as part of a conventional critiquing-based recommender system and we will go on to demonstrate the potential of these experiences to significantly improve the efficiency on standard unit critiquing.



**Fig. 2.** Each recommendation session is made up of a sequence of recommendation-critique pairs. Each recommendation pair is comprised of a recommended item and an individual critique (feature, value, type) applied to that item.

### 3.1 Recommendation Sessions

In a typical critiquing session the user will start with a high-level understanding of their needs. For example, when choosing a restaurant they might start by indicating a price-range and a location. During the course of a session this will be refined, as the user critiques the features of recommended restaurants, perhaps indicating that they are looking for somewhere that is less formal but more expensive than earlier recommendations. Thus, during a particular critiquing session a user may provide feedback on a range of different features.

We can model each recommendation session,  $s_i$ , as a sequence of *recommendation critique pairs*, as shown in Figure 2 and Equations 1-2; each  $r_i$  represents a recommendation and  $c_i$  is the critique that is applied by the user to that recommendation. Each  $c_i$  is represented as a triple,  $(f_i, v_i, type_i)$ , where  $f_i$  refers to the feature  $f_i \in r_i$  that is the focus of the critique,  $v_i$  is the value of  $f_i$  in  $r_i$  ( $r_i.f_i$ ), and  $type_i$  is the type of critique that is applied (typically,  $type_i \in \{<, >, =, <>\}$ ); see Equation 4. For now we can assume that each session terminates (see Equation 3) when the user chooses to *accept* a recommendation, indicating that they are satisfied with the recommendation, or when they choose to *stop* a given session, presumably because they have grown frustrated with the recommendations received. Thus we can add *accept* and *stop* to the set of permissible critique types such that every session terminates with one or other of these types.

$$s_i = \{p_1, \dots, p_n\} \tag{1}$$

$$p_i = (r_i, c_i) \tag{2}$$

$$terminal(s_i) = p_n = (r_n, c_n) \tag{3}$$

$$c_i = (f_i, v_i, type) \tag{4}$$

In general, the many users of a given critiquing-based recommender system will produce a large collection of critiquing sessions ( $S = \{s_1, \dots, s_k\}$ ) as they engage with the recommender system. The sessions reflect the *experience* of these

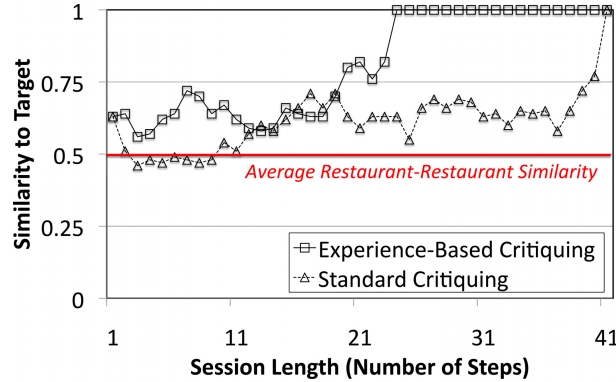


Fig. 3. Example session showing mean similarity to the target per critique step

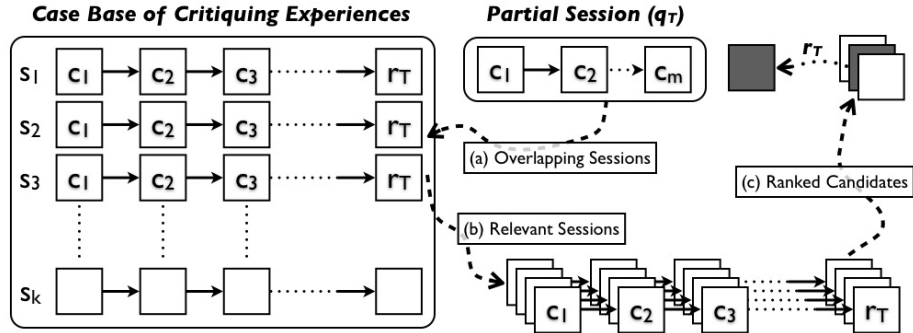
users and capture potentially useful knowledge about their preferences and the different trade-offs they tend to make. In this paper, we are interested in the potential for these experiences to inform the recommendation process itself. In other words, these critiquing sessions are the cases in a case base of critiquing experiences. For the remainder of this paper we will assume that only *successful sessions* — that is, those sessions where the user *accepts* a recommendation — are stored in the case base. Then we can treat this accepted recommendation as the *solution* of the case and the critique pairs that proceed it as the *specification* part. We will describe how to harness these critiquing experiences to improve the efficiency of the recommendation process by using a modified critique-based recommendation algorithm, called *experienced-based critiquing*, which differs from the traditional approach to critiquing in the manner in which new recommendations are generated; see Figure 4 for a brief overview.

### 3.2 Conventional Critiquing

According to conventional critiquing, when a user applies a critique  $c_i$  to an item  $r_i$ , the recommender responds by retrieving an item,  $r_T$ , which is compatible with  $c_i$ , in the sense that the item satisfies the critique  $c_i$ , and which is maximally similar to  $r_i$ , as in Equations 5-6<sup>1</sup>. Note that the form  $r.f$  indicates the value of feature  $f$  in recommended item  $r$  and  $apply(type, u, v)$  is true if and only if the predicate denoted by  $type$  is satisfied by the values  $u$  and  $v$ ; for example,  $apply(<, 25, 40)$  is *true* whereas  $apply(=, casual, formal)$  is not.

$$r_T = Recommend(r_i, c_i) = \underset{\forall r_j \in items \wedge satisfies(c_i, r_j)}{argmax} \left( sim(r_i, r_j) \right) \quad (5)$$

<sup>1</sup> Implementations will differ on issues such as the similarity metric used and also on whether items which have already been critiqued in a session are available for repeat recommendation.



**Fig. 4.** Experience-based critiquing reuses past successful sessions to identify terminal items that have proven popular in sequences of critiques that are similar to the user’s critiques. These items are a source recommendation candidates for the current user session.

$$satisfies(c_i, r_j) \leftrightarrow apply(type_i, r_j.f_i, r_i.f_i) \tag{6}$$

Figure 3 shows an example session profile for a standard critiquing session from the evaluation in Section 4. For each step in the session, we note the similarity of the recommended item (in this case a restaurant) to the target, which is known to the evaluation below. Here the session takes 41 steps to reach the target. Notice too how the similarity to the target does not rise monotonically as the session progresses. Often the user will select a critique but the new recommendation will be less similar to the target item than the critiqued case; the critiqued feature may be a better match to the target but this often comes at the expense of other features. In fact, notice how in the early part of the session (steps 3-9 inclusive) the similarity to the target dips below the average item-item similarity in our restaurant dataset; during these steps a random restaurant would likely have been a better overall match to the target than the recommended one.

### 3.3 Harnessing Critiquing Experiences

Experience-based critiquing extends conventional critiquing by reusing past sessions to guide the critiquing process. Instead of retrieving a new item that is maximally similar to the current recommendation, and compatible with the user’s critique, we recommend one of the items that past users have ended up purchasing/accepting under similar critiquing scenarios. This can be best understood in terms of three basic steps: (1) building experience cases from past users’ sessions; (2) identifying past critiquing sessions that are similar (i.e., relevant) to the current user session; (3) ranking recommendation candidates from the terminal items of these similar sessions.

**Building Experience Cases:** Before a case base of previous users’ critiquing experiences can be used in the recommendation process, the experience cases

must be built from the historical recommendation sessions (see Figure 2). This process is not as simple as extracting the critiques from the recommendation-critique pairs since in many scenarios users are liable to change their mind mid-session and as a result sequences of critiques can be in conflict [12]. For example, a user might start by looking for a product that is *cheaper than \$100* only to later shift towards looking for a product in the \$100 - \$150 range, once they start to recognise the different tradeoffs that exist in the target product space, and so eliminating recommendation candidates in the \$100 - \$150 range, based on the earlier critique, would be inappropriate. Accordingly we edit the current user's critiques by working backwards through the session starting with the last critique before target acceptance. If the current critique conflicts with a less recent critique (that has already been processed) then it is eliminated. This leaves a set of core critiques which represent the *boundaries* of the user's preferences with respect to the features that have been critiqued. Once this process is complete we also extract the final case as accepted by the user upon completion of their recommendation session. This case will serve as the candidate target for this particular critiquing experience (see Figure 4).

**Identifying Relevant Critiquing Sessions:** When a user applies a critique  $c_i$  to a recommended item  $r_m$  we will use the user's current (partial) critique session,  $c_1, \dots, c_m$ , as a query ( $q_T$ ), over the case base of past critique sessions, in order to identify a set of *relevant sessions*; see (a) and (b) in Figure 4. Briefly, a relevant session is one which has at least some overlap with the current query (see Equation 8), based on a particular overlap metric. In this case, we propose the simple overlap score shown in Equation 7, which computes the square of the number of critiques in  $q_T$  that are also present in a given session; the use of the square function here introduces a strong bias in favour of greater overlaps. Note that in the case that there are no relevant sessions, and thus no candidates to recommend, then we revert to standard critiquing and retrieve a new item that is maximally similar to the current recommendation and compatible with the user's critique.

$$OverlapScore(q_T, s_i) = \left[ \sum_{c_i \in q_T} \sum_{c_j \in s_i} match(c_i, c_j) \right]^2 \quad (7)$$

$$S^{REL} = RelevantSessions(q_T, S) = \left\{ s_i \in S : OverlapScore(q_T, s_i) > t \right\} \quad (8)$$

**Ranking Recommendation Candidates:** These relevant sessions ( $S^{REL}$ ) correspond to sequences of critiques that have previously led a user to a successful outcome. Each relevant session will terminate with some final, accepted recommendation case. The final recommendation case from the experience session with the largest overlap score intuitively makes a good recommendation for the current user (as shown in Equation 9). The recommended case,  $r_T$ , will be compatible with the last critique made by the user, will have overlapping critiques (the minimum amount of overlap depending on the threshold,  $t$ ) and will have been previously selected as a final recommendation.

$$r_T = \text{Recommend}(S^{REL}) = \underset{\forall s_i \in S^{REL}}{\text{argmax}} \left( \text{OverlapScore}(q_T, s_i) \right) \quad (9)$$

In summary then, when a user critiques an item, instead of just presenting that most similar remaining item that satisfies the critique, the experience-based critiquing technique recommends an item which has been frequently accepted by past users under similar critiquing conditions. This recommended item is usually not the most similar item to the critiqued item, allowing the recommender system to take larger steps through the product space and, hopefully, improve overall recommendation efficiency. For instance, returning to Figure 3, we also show the session profile for the corresponding experience-based critiquing session (with a case base of almost 3000 session cases and  $t = 15$ ). In this example case, the session reaches the target item at step 24, a 41% reduction in session length compared to the standard critiquing approach. Moreover, by and large, the similarities of the individual recommendations are typically 25% more similar to the target item than their corresponding recommendation from the standard critiquing session, and they never fall below the average item-item similarity level.

## 4 Evaluation

In conventional critique-based recommendation systems, new recommendations are influenced by the sequence of critiques in the current session. These critiques help to focus the recommender within the recommendation space. According to the new technique presented in this paper, the critiquing experiences of other users can also play a role in guiding the session. We evaluate this new technique, by comparing it to conventional critiquing. To do this we have developed a restaurant recommender system, based on a comprehensive database of Dublin restaurants.

### 4.1 Datasets

There are two key datasets used in this evaluation: *restaurants* (recommendation items) and critiquing *experience sessions*. The former stems from a recent crawl of an online restaurant database for Dublin. A total of 632 individual restaurants have been extracted and each is represented by 28 different features (e.g. price, quality, etc.), including 2 nominal, 7 numeric, and 19 binary features.

Ideally, we would like to be able to evaluate experience-based critiquing using real users. However, this is not currently feasible since it would require a major live deployment over an extended period of time. In the alternative, we adopt the approach taken by [7,11] to automatically generate experience critiquing sessions based on the behaviour of rational users, using the standard approach to critiquing described in Section 3.2. We do this by selecting a random restaurant as our *target*. From this target we automatically create a *query*, by selecting 4-8 features from the target at random, which acts as a starting point for each session. Each session begins by the recommender retrieving a best-matching

restaurant for the query. From here the ‘user’ must select a feature to critique. To do this we automatically select one of the features of the recommended restaurant and critique it *in the direction* of the target restaurant. For example, if the target restaurant has a price range of €20-€30 and the retrieved case has a price range of €40-€50, then if the price feature is selected for critiquing we will apply the *cheaper* (<) critique. Moreover, features are selected for critiquing based on a probability model that favours nominal and numeric features over binary features to simulate a more realistic critiquing session. Each session terminates once the target case has been recommended. We can repeat this process to generate an arbitrary number of critiquing sessions. In the case of this experiment, we generate several different queries for each of the 632 restaurant cases to generate a total of 2928 distinct critiquing sessions. These sessions range in length from 8-15 steps, with an average length of 11.42, and on average each session involves a critique of about 10 unique features. These sessions (or a random selection of these sessions) can then be used as the case base for our experience-based critiquing technique.

## 4.2 Algorithms and Methodology

We are interested in comparing the performance of a *standard* critiquing-based recommendation algorithm (as described in Section 3.2) to our *experience*-based algorithm. We generate a separate set of 500 target problems as our *test set* by using the aforementioned technique to generate a query-target pair. Next, we ‘solve’ each target problem, simulating the actions of a rational user using: (a) standard critiquing; and (b) the experience-based approach. In the case of the latter we use different sized case bases and overlap thresholds. A target problem is deemed to be solved once the problem’s target restaurant has been recommended, at which point we note the session length.

## 4.3 Results

The key performance issue to consider is whether the experience-based critiquing technique leads to earlier target recommendations, when compared to standard critiquing, and thus shorter sessions? If it does then this can lead to tangible benefits both for the user and for the recommendation service provider, since all other things being equal, shorter sessions mean less effort for the user and improved conversion rate for the service provider. To examine this, Figure 5 presents the average session length (across the 500 unseen test problems) for the standard critiquing and experience-based methods. We show the performance variation across different case base sizes, ranging from 500 cases to the full set of 2928 cases; the performance of the standard critiquing method presents as a dashed straight line since it is unaffected by case base size. We differentiate the performance of the experience-based method for different overlap thresholds (see Equation 8) in order to understand the benefits, or otherwise, of more strict thresholds. We ran simulations from  $t = 0$  (must have at least one overlapping critique) to  $t = 48$  (must have at least seven overlapping critiques).

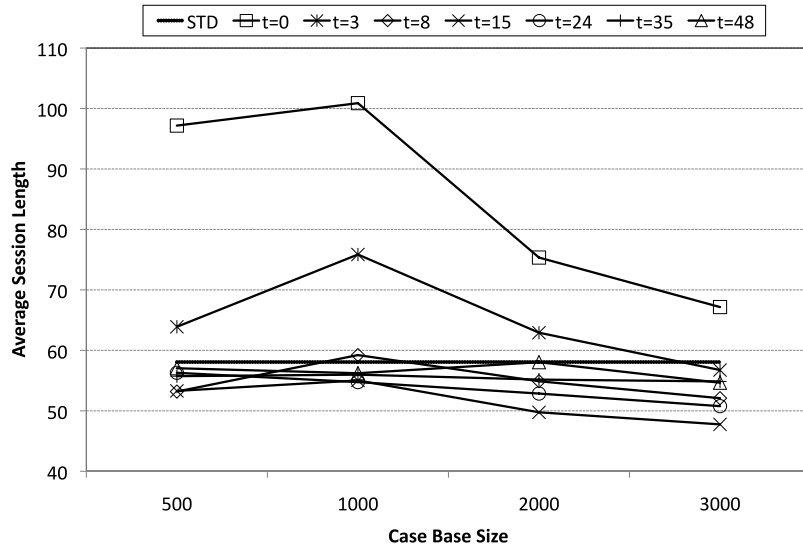
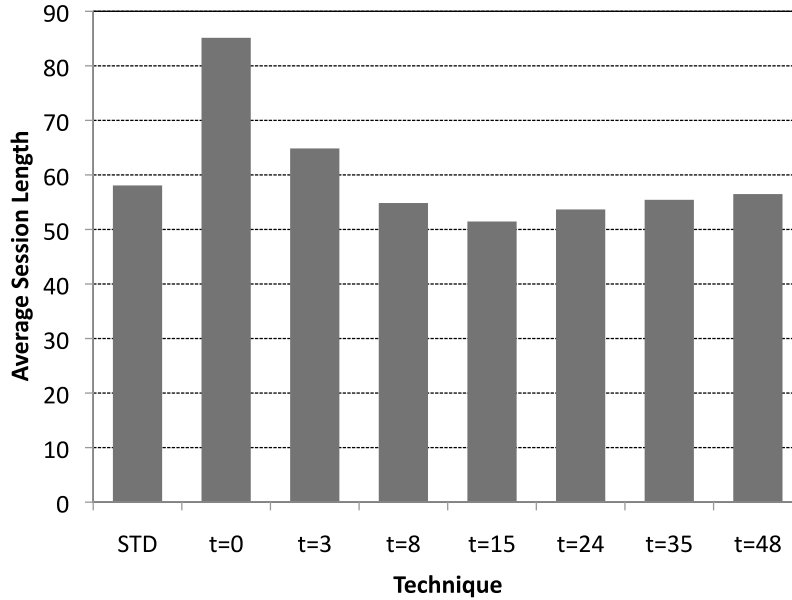


Fig. 5. The average session length results

The first point to notice is the performance of the standard approach to critiquing. On average its sessions extend to 58 steps. The experience-based critiquing techniques outperform the standard approach in some cases. Mostly these occur when a larger case base of experiences is available and when an appropriate threshold is chosen.

Across the various case base sizes results generally improve (except for case base with 1000 sessions). This is what is expected as larger case bases give more opportunity to find larger relevant overlapping experiences which leads to better case recommendation. Each individual case base (500, 1000, 2000) is made up of a random selection from the larger 3000 case-base. The sessions selected for the 1000 case base perform worse than those of 500 case base, even though the test problem coverage was 92% and 68% respectively for the two case bases. An examination of best practices for selecting previous sessions for experience cases is left for future work.

When we examine the various threshold settings for experience-based critiquing, we find that thresholds of 0 and 3 (at least one and two overlaps respectively) can not better the results of standard critiquing. Sessions with small overlaps do not generate beneficial recommendation candidates. When thresholds of 8 and greater (must have at least 3 overlapping critiques) are employed the average session lengths are lower than standard across the various case base sizes. Generally the trends improve as case base size increases. At the largest experience case base size,  $t = 15$  out performs standard by over 10 cycles. This reduction of 18% is seen as a good starting point for future experience-based critiquing research.



**Fig. 6.** Session lengths averaged across various CB sizes

For a clearer picture of the impact of varying  $t$  on session length, we have combined the various experience case base size results and presented the average session length across all of the techniques employed (Figure 6). As expected the value of  $t$  has a large effect on average session results. When  $t$  is low (at 0 and 3 - corresponding to at least 1 and 2 overlaps respectively) targets are recommended to the user that are not necessarily helpful in the long run and this leads to more protracted session lengths. However, as the threshold is made larger, the average session lengths improve and start to outperform standard critiquing. As the values of  $t$  are increased beyond  $t = 24$  the average session length also begins to increase. There is a tipping point where the minimum number of overlaps acceptable starts to affect the performance of the recommender. Figure 6 shows that for this simulation dataset, recommending items from experience cases which have between 3 and 5 overlapping critiques with the current session, offer the most benefit.

#### 4.4 Discussion

These results indicate that the experience-based critiquing can enjoy benefits compared to conventional critiquing. When we set the threshold to about four overlaps, our experience-based approach can lead to sessions that are shorter than those produced by standard critiquing. This means a cost saving for users, bringing them to the target recommendation in less steps than needed by conventional critiquing.

It is worthwhile to compare this new approach to critiquing to other approaches which aim to reduce critiquing effort. In Section 2 we described *compound critiquing*, designed to short-cut sessions by presenting users with dynamically generated compound critiques. These techniques do reduce session length but they also carry an extra overhead for the user in the form of more complex critiques to review. Indeed, in live-user trials a significant percentage of users tended to ignore compound critiques [6]. A key benefit of our new approach is that it can potentially deliver session length reductions but without introducing any new interface elements, such as extra compound critiques, and hence these benefits are likely to be available to all users and not just a fraction.

These results are based on artificial user data. Every effort has been made to ensure that these sessions are plausible — by following an offline evaluation protocol that has been successfully used in the past [7] — but they have not been generated by real users. Thus, these results should be considered as *preliminary* and need to be verified on live users. Nevertheless, the results do speak to the *potential* for experience-based critiquing to improve recommendation efficiency.

## 5 Conclusions

Critiquing-based recommendation techniques are useful when it comes to helping users to navigate complex product spaces. However, they can lead to protracted sessions and a high session-failure rate for end-users. While conventional approaches have been extended to deliver more efficient recommendation sessions (e.g. [7,11,13]), these extensions typically introduce an additional cost to the user, often in the form of a more complex interface and/or feedback options. Our goal in this work has been to improve the efficiency of critiquing-based recommender systems, but without introducing additional interface components and/or costs for the end-user. To this end we have described a novel critiquing strategy, which reuses a case base of prior critiquing experiences on the grounds that these past experiences are liable to encode important users preferences and feature trade-offs that may help to improve recommendation efficiency. We have described how these past experiences can be used to influence recommendation generation and the results of an offline evaluation demonstrate the potential benefits of this experience-based recommendation approach.

Future work will focus on a number of important areas, including the examination of experience case base session generation and selection, as well as new overlap scoring mechanisms. Also, a more comprehensive evaluation of the experience-based critiquing approach, involving live-users in a realistic online recommendation scenario is needed. In addition, the technique we have presented here has been based on *successful critiquing sessions* only; that is, by design the critiquing sessions stored in the case base all represent successful recommendation sessions, where the user eventually reached their target restaurant. In reality there is the potential to include failed sessions — where a user failed to get to an acceptable restaurant — as an additional source of critiquing experience, and in the future we will consider how these experiences can also be used to bias recommendation.

**Acknowledgement.** This work is supported by Science Foundation Ireland grant 07/CE/I1147.

## References

1. Bennett, J., Lanning, S.: The Netflix Prize. In: Proceedings of the KDD Cup and Workshop (2007)
2. Bridge, D.: Product Recommendation Systems: A New Direction. In: Aha, D.W., Watson, I. (eds.) ICCBR 2001. LNCS (LNAI), vol. 2080. Springer, Heidelberg (2001)
3. Burke, R., Hammond, K., Young, B.: The FindMe Approach to Assisted Browsing. *Journal of IEEE Expert* 12(4), 32–40 (1997)
4. Konstan, J.A., Miller, B.N., Maltz, D., Herlocker, J.L., Gordon, L.R., Riedl, J.: GroupLens: Applying Collaborative Filtering to Usenet News. *Communications of the ACM* 40(3), 77–87 (1997)
5. Linden, G., Hanks, S., Lesh, N.: Interactive assessment of user preference models: The Automated Travel Assistant. In: Jameson, A., Tasso, C.P., C. (eds.) *User Modeling: Proceedings of the Sixth International Conference*, pp. 67–78. Springer, Wien (1997)
6. McCarthy, K., McGinty, L., Smyth, B., Reilly, J.: On the evaluation of dynamic critiquing: A large-scale user study. In: Veloso, M., Kambhampati, S. (eds.) *Proceedings of the Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference (AAAI-2005)*, pp. 535–540. AAAI Press / The MIT Press (2005)
7. McCarthy, K., Reilly, J., McGinty, L., Smyth, B.: On the dynamic generation of compound critiques in conversational recommender systems. In: De Bra, P.M.E., Nejdl, W. (eds.) *AH 2004. LNCS*, vol. 3137, pp. 176–184. Springer, Heidelberg (2004)
8. McGinty, L., Smyth, B.: Comparison-Based Recommendation. In: Craw, S., Preece, A.D. (eds.) *ECCBR 2002. LNCS (LNAI)*, vol. 2416, pp. 575–589. Springer, Heidelberg (2002)
9. McSherry, D.: Incremental Relaxation of Unsuccessful Queries. In: Funk, P., González Calero, P.A. (eds.) *ECCBR 2004. LNCS (LNAI)*, vol. 3155, pp. 331–345. Springer, Heidelberg (2004)
10. Pu, P., Faltings, B.: Decision Tradeoff Using Example-Critiquing and Constraint Programming. *Constraints* 9(4), 289–310 (2004)
11. Reilly, J., McCarthy, K., McGinty, L., Smyth, B.: Dynamic critiquing. In: Funk, P., González Calero, P.A. (eds.) *ECCBR 2004. LNCS (LNAI)*, vol. 3155, pp. 763–777. Springer, Heidelberg (2004)
12. Reilly, J., McCarthy, K., McGinty, L., Smyth, B.: Incremental critiquing. *Knowledge-Based Systems* 18(4-5) (2005)
13. Reilly, J., Zhang, J., McGinty, L., Pu, P., Smyth, B.: A comparison of two compound critiquing systems. In: *IUI 2007: Proceedings of the 12th international conference on Intelligent user interfaces*, pp. 317–320. ACM Press, New York (2007)
14. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: GroupLens: An open architecture for collaborative filtering of netnews. In: *Proceedings of ACM Conference on Computer-Supported Cooperative Work (CSCW 1994)*, August 1994, pp. 175–186. ACM Press, North Carolina (1994)

15. Shardanand, U., Maes, P.: Social Information Filtering: Algorithms for Automating "word of mouth". In: Proceedings of the SIGCHI Conference on Human factors in Computing Systems (CHI 1995), pp. 210–217. ACM Press/Addison-Wesley Publishing Co., Denver (1995)
16. 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 2001), pp. 1443–1448. Morgan Kaufmann, San Francisco (2001)
17. Smyth, B., McGinty, L.: An Analysis of Feedback Strategies in Conversational Recommender Systems. In: Cunningham, P. (ed.) Proceedings of the Fourteenth National Conference on Artificial Intelligence and Cognitive Science, AICS 2003 (2003)
18. Yang, Q., Zhang, H.H., Li, T.: Mining web logs for prediction models in www caching and prefetching. In: KDD 2001: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 473–478. ACM Press, New York (2001)
19. Zhang, J., Pu, P.: A comparative study of compound critique generation in conversational recommender systems. In: Wade, V.P., Ashman, H., Smyth, B. (eds.) AH 2006. LNCS, vol. 4018, pp. 234–243. Springer, Heidelberg (2006)