Wolfgang Wörndl and Béatrice Lamche

User Interaction with Context-aware Recommender Systems on Smartphones

Benutzerinteraktion mit kontextsensitiven Empfehlungssystemen auf Smartphones

Keywords: recommender systems, user interfaces, context, mobile applications

Summary. In this article we give an overview on selected aspects of user interaction with context-aware recommender systems on smartphones. We discuss these according to the three steps of user interaction with recommender systems using subjective and objective evaluation criteria: 1. Preference elicitation: how input methods on mobile devices can influence the users’ rating behavior, 2. Result delivery and presentation: how results can be adapted to the mobile context, 3. Feedback, critiquing and refinement: how interactive explanation can improve the user experience. The selection of examples is based on several studies we did in different mobile scenarios.


1. Introduction

Recommender systems recommend movies, restaurants or other relevant items to an active user or user group. Recommender systems rely on ratings of items by users (collaborative filtering) or a variety of information about users and items, e.g. matching item metadata with user preferences in a content-based filter. Recently, the focus in recommender systems research has been shifting from investigating the underlying algorithms to studying the user experience (Konstan and Riedl, 2012). This implies the delivery of the recommendations to the user in an intuitive and enjoyable way.

In mobile scenarios, i.e. users with smartphones or other mobile devices, generating accurate recommendations can help reducing the omnipresent information overload. Mobile recommender systems can also support users that do not exactly know what they want or need in their exploratory search process. However, mobile information access suffers from limited resources regarding input capabilities, smaller display sizes and other restrictions of small mobile devices. Therefore, the user interface for a mobile recommender system cannot just be transferred from a desktop setting, but has to be tailored to the specific properties of mobile devices. Cognitive load and limited attention spans of users while moving also add to the need of adapted information access.

Mobile recommender systems face challenges (Ricci 2011), especially the context of the interaction has to be taken into account. The goal of this article is to provide an overview of user interaction with mobile recommender systems including contextualization. We outline selected issues, which is based on several studies we did to test novel interaction methods in different mobile scenarios. Mobile device in this work means smartphone or a similar device with a smaller screen.

The contribution is organized along three steps of user interaction with recommender systems:
1. We investigate the influence of input methods for preference elicitation (Section 3)
2. We introduce a model for proactivity and explain a study on contextualization (Section 4)
3. We describe interactive explanations for improving user feedback and critiquing (Section 5)
First, we introduce mobile recommender systems and explain background information in Section 2.

2. Background

Mobile Recommender Systems
According to (Ricci 2011), mobile recommender systems meet several challenges. These include user interface design, context-awareness, explanations of recommendations, distributed computing models and proactivity. While this article focuses on user interface issues and context-awareness, the other challenges should also be taken into account. Context-awareness refers to an adaptation of the system interaction to the current situation of the user. We can distinguish several types of (mobile) context including:

- Physical context: position and time but also light, temperature and other data from a mobile device's sensors
- Cognitive or modal context: e.g. the user's goals, current activity and state of mind
- Social context: whether the user is alone or the presence and role of other people around the active user

Traditional recommender systems usually follow a request-response pattern, i.e. these systems only return item suggestions when a user makes an explicit request (pull or "reactive system"). Proactivity means that the system pushes recommendations to the user when the current situation seems appropriate, without explicit user request (Wörrdl, Huebner, Bader and Gallego-Vico 2011). Proactive recommendations seem conceivable in mobile scenarios, for example supporting a tourist visiting a city with timely recommendations for nearby points-of-interests.

Interaction with Recommender Systems
We can identify three steps for user interaction with recommender systems:
1. Preference elicitation: Users either explicitly state their preferences, or their behavior is implicitly observed and user models are derived. Often, ratings of a user for an item are used as input for collaborative filtering recommender systems.
2. Result delivery and presentation: After computing a list of recommended items, the results have to be delivered and communicated to the user on the mobile device.
3. Feedback, critiquing and refinement: Users can give feedback on the recommended items to allow for refining the result list or select an item.

(Pu, Faltings, Chen, Zhang and Viappiani 2011) propose a similar interaction model but include a fourth step, the user picking his or her final choice. The difference between step 1 and 3 is, that in step 1, a user states his or her initial preferences by e.g. rating an item he or she already knows. In step 3, the user gives feedback on the result of the recommendation process by judging the proposed items and refining the search process by stating "similar item, but cheaper" in a conversational recommender system, for example (Chen and Pu 2012). A conversational recommender system conducts a dialog with the user for feedback, in this case step 3 and 2 are iterated. The feedback step can be done by critiquing proposed items, submitting ratings or modifying their original preferences. Critique-based recommendation can be particularly effective in supporting mobile users in product selection decisions, as (Ricci and Nguyen 2007) have shown. Often, users are better at critiquing presented alternatives than at specifying what they want (Konstan and Riedl, 2012).

User-Centric Evaluation
While traditionally recommender systems have been concentrating on accuracy, the focus has been shifting to a broader set of measures and a more user centric evaluation (Konstan and Riedl, 2012). (Shani and Gunawardana 2011) distinguish between offline experiments, user studies and online experiments. In offline experiments, a pre-collected dataset is used to answer questions about the accuracy and predictive power of algorithms, but this approach cannot really evaluate how a user in a certain situation perceives a recommendation. Online experiments measure the change in user behavior when interacting with different recommendation systems or algorithm, for example users of which variant select suitable recommendations more often. However, it is hard to test why users made these decisions in online experiments.
Therefore, user studies are often employed to evaluate recommender systems from a user’s perspective. A user study is conducted by asking test subjects to interact with the recommender system and collect quantitative and qualitative data. Quantitative data is sometimes referred to as objective data and can include the time to complete a specific task, the number of interaction steps and accuracy of tasks or recommendations. Qualitative or subjective data is often measured using questionnaires the participants have to fill out before, during or after the study. By doing so, the researcher can acquire data about user satisfaction, perceived recommendation accuracy or information that is difficult to measure otherwise, such as a subject’s state of mind.

The following sections include examples for utilizing subjective and objective criteria for evaluation with user studies in the three introduced steps of user interaction with recommender systems in different mobile scenarios.

3. Preference Elicitation

The first interaction step is preference elicitation. Techniques for eliciting user preferences for recommender systems can be classified into explicit and implicit methods (Jawaheer, Weller and Kostkova, 2014). In this section, we focus on explicit methods and explain two studies investigating input methods for mobile recommenders in Section 3.2.

3.1 User Profile Acquisition

Explicit vs. Implicit Acquisition

Traditionally, explicit methods to express the user’s preferences methods have been dominant, in particular rating items on a certain scale. We can distinguish between unary (e.g. „like“ on Facebook or „product was purchased“), binary (e.g. „good/bad“ or „thumbs-up/-down“) and scalar (e.g. numerical scale 1-5) rating scales. It has been shown in re-rating experiments that ratings are not always reliable and consistent but may depend on presentation issues or other factors (Jawaheer, Weller and Kostkova, 2014). Other means for explicit user feedback can be used, e.g. letting users write product comments or applying tags in social media applications, or having the user select an appropriate stereotype to speed up the initial preference acquisition phase (Lamche, Pollok, Wördl and Groh 2014).

Implicit user feedback is based on observable behaviors exhibited by users (Jawaheer, Weller and Kostkova 2014). An example for a factor for user behavior is the time spent to consume an item, e.g. on a Web page. Implicit methods are less intrusive but may suffer from poorer quality in terms of expressing users’ preferences and less control for users. Some approaches have proposed to convert implicitly acquired data into explicit ratings. Implicit methods have been used less than explicit methods, but this may mostly be due to the lack of datasets with corresponding data. (Jawaheer, Weller and Kostkova, 2014) point out several different characteristics in the properties of explicit and implicit user feedback. From the mobile perspective, the most important one is the cognitive effort for users. While implicit user feedback does in general not interrupt or disturb the user, explicit feedback requires some action by the user, which may be cumbersome in a mobile environment. An example for implicit feedback on mobile devices is how much time a user spends at a particular point-of-interest.

Input Methods on Mobile Devices

The available input methods on mobile devices have to be taken into account for eliciting explicit user preferences. There are several principle methods to interact with a mobile device with a touchscreen (e.g. a smartphone). The most common forms are on-screen user interface elements such as buttons or sliders and allow users to interact by tapping on the screen. A swipe (or flick) gesture is a movement of one finger in one direction across the screen and can be used to delete an item from a list, for example. Free-form gestures do not require the user to actively touch the screen but to move the device to initiate functions (Wördl, Weicker and Lamche 2013). For example, shaking the device is sometimes used to select a new random track in a music player.

Additional capabilities of today’s smartphones can be utilized to simplify a search for recommended items. One option is to take a picture of an item as a starting point for the exploratory search process. This is also useful when a keyword-based search is not enough to describe what the user is looking for. In the example in Fig. 1, the user first takes a picture of a t-shirt. Then, the system applies image recognition techniques to extract discriminating features, compares them to a dataset of items and selects similar products to the user as recommendations (Fig. 1, right).
The process to elicit user preferences does not only influence the user experience but input methods may alter the actual preferences or ratings the user submits to the system and thus ultimately change the outcome of the recommendation process and overall effectiveness of the system. We explain an example in the next subsection.

3.2 Input Methods for Submitting Ratings

Smartphones and other mobile devices offer different methods to interact with applications. Mobile users may prefer one input method to another for recommender functionalities such as rating an item. In addition, explicitly entered information may contain some level of noise. Usually, recommender systems research assumes stable ratings, i.e. the assumption is that an available rating exactly reflects the user's opinion about an item. Therefore we have examined the usage of mobile input methods for a recommender system in two studies.

In a first study, we have investigated the user acceptance of several input methods to rate an item and apply other functions of a mobile recommender (Wörndl, Weicker and Lamche 2013). The scenario is a movie search and recommendation application that resembles the Internet Movie Database (IMDb) mobile application. Users can perform different recommender functions such as finding new recommended items, selecting or dismissing an item from a list, or performing a rating (Fig. 2). We provided at least two different input methods for each recommender function, either on-screen buttons, menu options (the user has to select a specific "menu" option to show additional buttons), or gestural interface options. For example, we implemented a One-Finger-Hold Pinch two-finger gesture for rating. In this case, one finger rests on the screen, while a second finger moves on the screen to adjust the rating on the given scale. In a user study, we tested which input methods the participants preferred as a subjective measure (Wörndl, Weicker and Lamche 2013). It turned out that gesture usage depended on the specific recommender function. For example, users preferred the on-screen button for rating an item over the One-Finger-Hold Pinch gesture but would use the free form gesture Shaking the Device to retrieve a new (random) item from the recommendation list. In general, users preferred simpler gestures and rarely switched their input method for a function during the test. Omitting on-screen buttons was seen as an option in activities where content space is limited, in our case the overview list of items. The options menu was not very popular in any of the used cases.
In a second study, we have investigated the influence of the input methods on the resulting ratings as an example to objectively testing the effectiveness of user interaction methods (Najafian, Wörndl and Lamche 2014). 20 participants were first asked to choose and rate 16 movies manually. Then, the subjects had to re-rate the same items in the smartphone application using three input methods: 1. tapping on a 1 to 10 scale of stars (Fig. 2), 2. applying the explained One-Finger-Hold-Pinch gesture, 3. performing a Tilt free-form gesture. Tilting is performed by shifting the smartphone horizontally, shifting to the right increases the rating and shifting to the left decreases it. Afterwards, we calculated the errors in applying the ratings based on the users’ initial ratings. The study investigated two context scenarios. The first one was conducted while the users were sitting and concentrated on the task. In the second scenario, the users were walking around and thus not fully concentrated. We evaluated the error for every rating method by calculating the root mean squared error (RMSE), i.e. the squared deviation between the intended and actual rating, as an objective measure. Fig. 3 shows the normalized root mean squared error percentage for the three methods and two scenarios (Najafian, Wörndl and Lamche 2014). As expected, the outcome in the sitting scenario is more precise than in the walking scenario, regardless of which interaction method has been used. Among the three input methods, the on-screen button has the lowest error (with less than 3% were close to the intended rating), the touchscreen gesture has a medium accuracy and the tilt gesture has the highest NRMSE% (more than 10%) in our study. Thus, the input method has a considerable effect on the resulting rating behavior and noise.

Fig. 2 Movie details and rating interface

Fig. 3 NRMSE% results
4. Delivering Recommendation Results

After user preferences are explicitly entered and/or implicitly derived, the recommender system calculates a list of recommended items and communicates it to the user. To do so, multiple options exist. We discuss some aspects with regard to mobile interaction in this section.

(Ricci 2011) suggests that map-based interfaces can be utilized to present recommended items such as points-of-interests. This is suitable for data with geo-location but not for all application domains. Because of limited screen space of devices and restricted attention spans of users, applications have to limit textual information and rely more on visual presentation in form of pictures. A simple strategy is to show only one item at a time (Pu, Faltings, Chen, Zhang and Viappiani 2011). For more results, a grid view based on item images and less text is an alternative to the typical list view of recommender results (see mobile shopping examples below). Explanations are also very important in the mobile scenario but are problematic to implement with small screens (Ricci 2011). We will outline an approach for interactive explanations on mobile devices in Section 5.

The actual presentation of recommendation results is largely application-dependent. We thus focus in this section on how to notify the mobile user in a proactive recommender system (4.1) and give an example how to contextualize the generation of recommendation results as important aspects of mobile recommendation delivery (4.2).

4.1 Notification Modes and Proactivity Model

Modalities

Section 2 already introduced proactivity as an alternative for the standard reactive interaction model that is suitable in a mobile setting. In a proactive recommender system, we need to notify the user about available recommendations. The general goal is to notify the user without interrupting him or her, and also not disturbing the environment. Situations with different levels of personal and social interruptability call for well-adjusted announcement stimuli, i.e. for different notification modalities. High sensitivity corresponds to low interruptability and calls for cues of moderate obtrusiveness, which have little potential of being disruptive. The design space for notifications on smartphones covers cues of three major modalities:

- Visual: Visual cues, e.g. textual messages, notification bars or flashing screens, are generally the least intrusive ones and are less likely to disturb the environment
- Tactile: Haptic or tactile cues, e.g. vibrations, have a greater efficacy than visual cues while still avoiding social obtrusiveness in most environments
- Acoustic: Auditory cues have the highest potential for raising awareness. The extent can be determined by adjusting volume, length, and frequency of the tone

Proactivity Model

To answer the question when the system should notify the mobile user about suitable recommendations, we developed a two-phase proactivity model (Wördl, Huebner, Bader and Gallego-Vico 2011). In the first phase, the system assesses whether or not the current situation warrants a recommendation. To do so, the system takes several attributes such as geographical, temporal, device and social context into account and calculates a score S1. If S1 exceeds a first threshold, the second phase will be initiated. If S1 = 1, the highest possible value, then a recommendation will be triggered in any case. If the current situation does not warrant a recommendation, no matter how high a particular item would score, S1 is set to 0 and the recommendation process is aborted without considering items for recommendation.

The second phase takes the suitability of particular items into account by calculating a second score S2 per item. S2 is computed by applying any contextual or non-contextual recommender algorithm, e.g. the normalized predicted rating of a collaborative filtering algorithm for an item. If the score of one or more items is above a second threshold, these items are considered good enough in the current context and communicated to the user. Thereby, the system can utilize one or more of the introduced notification modalities, which may depend on current device settings. For example using tactile and/or acoustic notifications only when an item has an unusually high score and the device is not in silent mode.
4.2 Contextualizing Recommendation Results

We developed and evaluated a context-based mobile shopping recommender system to find out if context-aware information such as weather, budget and shopping intent can predict the user's current shopping interest to improve accuracy and efficiency of a mobile shopping recommender system.

Solution Design

To integrate the contextual information into the recommender system, context-driven querying and -search was adopted. This approach uses contextual information and/or user's specified interest to query or search a repository of resources and present the most appropriate ones to the user. Corresponding to this approach, we applied a case-based recommendation technique. Each case in the case base is composed of an item and the contextual situation under which the item is bought. Here a contextual situation is a combination of several context factors and their corresponding values. A user query is composed of a logical query with fixed context constraints and a feature value vector of context factors and their corresponding value that the user wishes to be considered in the recommendations. For example, if a user is a budget buyer and wants to buy sportswear when the temperature is hot in opened stores nearby, the query may be structured as follows:

\[
\text{query} = \{ ((\text{distance}<2000m) \land \text{timeopen}=\text{now}+30\text{min}), (\text{budget(budget\_buyer)}, \text{intent(sports)}, \text{temperature(hot)}) \}\]

The system then searches the case base and selects the nine cases with the most similar context situation and recommends these items or similar ones to the user. However those items are not only ranked according to the level of similarity to the current context. Previous works have shown that diversity is an important consideration to ensure the coverage of the current scope of candidate items, in particular in exploratory scenarios. Thus, a bounded greedy selection algorithm (Lamche, Trottmann and Wörndl 2014) was extended to select the cases with the most diverse set of items among the retrieved most similar cases.

Case Model and Similarity Assessment

The case base consists of two components: the item bought (I) and the context situation (C): \(CB = I \times C\). Each case \(c = (i, e) \in CB\) in the case base is composed of two sub-elements \((i, e)\) which are instances of the spaces \(I, C\) respectively. The cases are not correlated with the user who submits it, thus the cases are not linked to the user model (Ricci, Arslan, Mirzadeh and Venturini 2002). A case is created when the user purchases the item. \(C\) is the data structure that defines the context situation under which the item is bought. It is composed of a feature value vector of context factors and their corresponding values that the user wishes to be considered and a feature value vector of context factors and their corresponding factor importance weights. The factor importance weights reflect the level of influence of the context factors on the recommendations of clothing items. They are determined by the type of clothes and have been calculated in the experiment introduced before. An example for the feature value vector for a budget buyer who is looking for sports clothes when the temperature is hot could be:

\[
\text{context}\_\text{attributes}=\{(\text{budget(budget\_buyer)}, \text{intent(sports)}, \text{temperature(hot)}), (\text{w\_budget}(0.7), \text{w\_intent}(0.6), \text{w\_temperature}(0.9))\}.
\]

\(I\) is the data structure that describes the clothing item bought by the user. It is represented as a feature value weight vector (Lamche, Trottmann and Wörndl 2014). To generate recommendations, cases with context situations similar to the user's current context can be retrieved and the items contained in those cases can be used directly for the recommendations. They can also be used as reference items to find other similar items to recommend. In order to get the similarity between the current context and retrieved cases, we applied the Heterogeneous Euclidean-Overlap Metric (HEOM) proposed by (Ricci, Arslan, Mirzadeh and Venturini 2002). This metric measures the distance between two weighting vectors of context factors.
Fig. 4 Grid view of set of items and detail view

Fig. 4 shows the user interface of our test application with contextual explanations in the detail view of items.

Evaluation

To acquire and integrate context relevance for a user study, we first conducted an online study to investigate how the influence of contextual factors change the user’s purchasing decision for clothing items and thus provides quantitative measurements that can be used as weighted attributes in the similarity measurement in the recommendation algorithm (Baltrunas, Ludwig and Ricci 2011). The main context factors in our model are: distance, day of the week, temperature, time available, transport, weather, time of the day, crowdedness, intent of purchasing, companion, season and budget.

23 people participated in the subsequent user study. The study was designed as within-subjects, one group of people tested both variants: The introduced context-aware system (CARS) and a baseline, which basically recommends items without eliciting preferences from the user but also uses a diversity-based approach to ensure the coverage of the recommended items (Lamche, Trottmann and Wörndl 2014). The users were asked to select the most appealing item while imagining her or himself being in a context scenario that was randomly selected by the system from a set of five pre-created context scenarios. A typical context description was: “Imagine that you want to buy clothes for daily wear, you are a budget buyer and the temperature is cold. You don’t care about the distance to the shops.” For each context scenario, about four different context factors were included. An overview of the evaluation results can be seen in Table 1. Next to the means of the two systems, the standard deviations are shown and the last column denotes the p-value of a one-tail paired t-test. As subjective criteria, the perceptive measures were gathered through a questionnaire listing statements that had to be rated on a five-point Likert scale (from 1, strongly disagree to 5, strongly agree with 3 being neutral). The time consumption was stopped in seconds and the number of critiquing cycles was automatically counted (objective criteria).

Results

<table>
<thead>
<tr>
<th></th>
<th>CARS mean</th>
<th>std dev</th>
<th>Baseline mean</th>
<th>std dev</th>
<th>p value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perceived Effort</td>
<td>3.74</td>
<td>0.92</td>
<td>3.61</td>
<td>1.16</td>
<td>0.3</td>
</tr>
<tr>
<td>Perceived Effectiveness</td>
<td>4.09</td>
<td>0.79</td>
<td>3.74</td>
<td>0.92</td>
<td>0.029</td>
</tr>
<tr>
<td>Critiquing Cycle</td>
<td>2.83</td>
<td>2.46</td>
<td>3.43</td>
<td>2.35</td>
<td>0.171</td>
</tr>
<tr>
<td>Time Consumption</td>
<td>122.91 s</td>
<td>77.67</td>
<td>117.52 s</td>
<td>73</td>
<td>0.405</td>
</tr>
<tr>
<td>System’s Preference</td>
<td>4.04</td>
<td>0.77</td>
<td>3.39</td>
<td>0.94</td>
<td>0.004</td>
</tr>
</tbody>
</table>

To measure the perceived effort of the system, the users were asked if it was easy to find the required information. The context-aware system slightly beats the baseline (3.74 vs. 3.61), but the difference is not significant (p = 0.3). To measure
the effectiveness of the system, the subjects were asked if the system was effective in helping to complete the scenario. The effectiveness of the context-aware system was rated significantly better than the baseline (p = 0.029 < 0.05). Users were also asked to rate how much they liked using these two systems. In Table 1, it can be seen that the mean rate of the context-aware system is higher than the one of the baseline by about 0.65 points and the difference is significant (with p = 0.004 < 0.05).

As far as objective criteria are concerned, the number of critiquing cycles of the context-aware recommender system is in average smaller compared to the baseline. The mean of critiquing cycles of the context-aware system (2.83 cycles) is also smaller than the one of the non-context-aware system (3.43 cycles), but the difference is not significant (p = 0.171). The non-context-aware recommender system slightly beats the context-aware system in terms of time consumption (117.52 vs. 122.91 seconds), but not significantly. However, the majority of time consumption of the context-aware variant is more stable and is neither too long nor too short.

In our user study, we were able to show that our developed context-aware system has a better performance regarding prediction accuracy and decision effort than a simpler baseline system. The context-aware system was also perceived as being significantly more effective in adapting recommendations to the users’ different context scenarios. Furthermore, users showed a clear preference of the context-aware variant and were able to understand the benefits of taking contextual information into account very well.

5. Feedback, Critiquing and Refinement

As a third interaction step, users can give feedback on the recommended items (critiquing) to allow for refining the result list and finally select an item. In Section 5.2 we explain our approach for interactive explanations to support user feedback in more detail.

5.1 Feedback in Mobile Recommender Systems

In a mobile setting, it is important to avoid lengthy initiation procedures but quickly present results and provide means for user feedback (step 3 instead of step 1 in our interaction model, cf. Section 2). (Ricci 2011) argues that critiquing and conversational recommender have some general advantages on the mobile domain. Criticizing a real product is convincing because the system first provides some immediate benefit to the user by showing recommendations, and then requests and motivates the user to reveal additional preferences in the form of critiques.

Users can express critique and feedback implicitly, if for example the user selects an item from a list of results and investigates it in more detail by looking at a details screen, may indicate some interests. If the user is watching a video or listening to an audio to the end can also be interpreted as an implicit positive rating. User feedback can also be given by natural language dialogue (Chen and Pu 2012). Current smartphones offer this option in principle but is not widely used at this time.

In a proactive recommender system, it is not only important to allow user feedback for items, but also on the point of time of the recommendation. To do so, we implemented and tested a “not now” button in a proactive restaurant recommender for Android smartphones (Gallego-Vico, Wörndl and Bader 2011). Thus, the user can state that the current time was not appropriate, regardless of how much he or she actually likes the recommended items. This (optional) feedback can be integrated in the first phase of our proactivity model (cf. Section 4.1).

5.2 Interactive Explanations

The already mentioned mobile shopping example already includes explanations and simple feedback options (thumbs-up/-down), but we extended this interaction model with interactive explanations. Explanations of recommendations help users to make better decisions in contrast to recommendations without explanations while also increasing the transparency between the system and the user. Therefore, we developed a concept featuring interactive explanations for mobile shopping recommender systems in the domain of fashion (Lamche, Adigüzel and Wörndl 2014). We integrated interactive explanations into a critiquing-based recommendation approach for a smartphone. A prototype, which generated personalized interactive explanations using the current state of the user’s inferred preferences and the mobile context was evaluated within a user
study. The main vision behind interactive explanations is to use them not only as a booster for transparency and understandability of the recommendation process but also as an enabler for user control.

Two types of explanations were defined: recommendation- and preference explanations (Fig. 5). Recommendation explanations are interactive textual explanations. They justify why an item in the recommendation set is relevant for the user and let the user make direct changes to the inferred preferences. The generation is based on the set of recommended items, the user model and the mobile context. Preference explanations allow the user to inspect the current state of the system’s understanding of the user’s preferences and to make direct changes to the preferences. Their generation is based on the user model.

We conducted a user study to compare the designed interactive explanation with a baseline of simple, non-interactive explanations. Overall, 30 people participated in the study. The task was to find an appealing clothing item for a special event within a specific clothing type, color and price range. To measure subjective results, we asked the participants to rate statements about transparency, user control, efficiency and satisfaction based on their experience with the system on a five-point Likert scale (from 1, strongly disagree to 5, strongly agree). After having tested both variants, participants stated which variant they preferred and why that was the case. In order to measure objective results, we tracked the time and the number of critiquing cycles until the user selected an appropriate clothing item. Results of the user study showed that the proposed concept performed significantly better compared to the approach with non-interactive simple explanations in terms of the user study’s main goals to increase transparency and scrutability, and its side goals to increase perceived efficiency and satisfaction. However, the baseline performed better regarding the time consumption, but not significantly. One reason could be that users spent more time with reading the more detailed explanations. Overall, the developed interactive explanations approach demonstrated the user appreciation of transparency and control over the recommendation process in a conversation-based mobile recommender system tailored to a modern smartphone platform (Lamche, Adigüzel and Wörndl 2014).

Fig. 5 Recommendation and preference explanations

6. Conclusion

In this article, we gave an overview on selected aspects of user interaction with context-aware recommender systems on smartphones or other mobile devices with smaller screens. We covered selected examples according to the three steps of user interaction with recommender systems using subjective and objective evaluation criteria:

1. Preference elicitation: users prefer different input methods for recommender functions. We have also shown that input methods can influence users rating behavior, i.e. potentially introduce noise.
2. Result delivery and presentation: contextualizing recommender results lead to better performance regarding prediction accuracy, decision effort and user satisfaction in our study.

3. Feedback, critiquing and refinement: introducing interactive explanations improved user feedback and critiquing options in our conversational mobile shopping recommender.

Overall, improving the user experience can further advance the adoption of mobile recommender systems. Future challenges include more work on implicit profile acquisition in the mobile scenario. Smartphones collect a lot of data through their sensors that can be potentially exploited for recommendations. For explicit user interaction, it would be interesting to see how more complex free-form gestures can be integrated in the user interaction process. In general, guidelines for user interfaces for mobile recommender systems could be developed. Finally, we tried contextualization and interactive explanations in the mobile shopping scenario but our future plans are also to test these and other methods such as proactive recommenders in different application scenarios, for example personal information management.

References


