Interactive Explanations in Mobile Shopping Recommender Systems

Béatrice Lamche
TU München
Boltzmannstr. 3
85748 Garching, Germany
lamche@in.tum.de

Uğur Adıgüzel
TU München
Boltzmannstr. 3
85748 Garching, Germany
adiguzel@in.tum.de

Wolfgang Wörndl
TU München
Boltzmannstr. 3
85748 Garching, Germany
woerndl@in.tum.de

ABSTRACT

This work presents a concept featuring interactive explanations for mobile shopping recommender systems in the domain of fashion. It combines previous research in explanations in recommender systems and critiquing systems. It is tailored to a modern smartphone platform, exploits the benefits of the mobile environment and incorporates a touch-based interface for convenient user input. Explanations have the potential to be more conversational when the user can change the system behavior by interacting with them. However, in traditional recommender systems, explanations are used for one-way communication only. We therefore design a system, which generates personalized interactive explanations using the current state of the user's inferred preferences and the mobile context. An Android application was developed and evaluated by following the proposed concept. The application proved itself to outperform the previous version without interactive and personalized explanations in terms of transparency, scrutability, perceived efficiency and user acceptance.

Categories and Subject Descriptors

H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*Interaction styles, User-centered design*

General Terms

Design, Experimentation, Human Factors.

Keywords

mobile recommender systems, explanations, user interaction, Active Learning, content-based, scrutability

1. INTRODUCTION

In today's world, we are constantly dealing with complex information spaces where we are often having trouble to either find what we want or make decisions. Mobile recommender systems are addressing this problem in a mobile environment by providing their users with potentially useful suggestions that can support their decisions to find what they are looking for or discover new interesting things. 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 [8]. However, recommender systems employing explanations so far did not leverage their interactivity aspect. Touch based interfaces in smartphones reduce user effort while giving input. This can empower the interactivity for explanations. There are two main goals of this work. One is to study whether a mobile recommender model with interactive explanations leads to more user control and transparency in critique-based mobile recommender systems. Second is to develop a strategy to generate interactive explanations in a content-based recommender system. A mobile shopping recommender system is chosen as application scenario. The rest of the paper is organized as follows. We first start off with some definitions relevant for explanations in recommender systems and summarize related work. The next section explains the reasoning behind and the path towards a final mobile application, detailing the vision guiding the process. The user study evaluating the developed system is discussed in section 4. We close by suggesting opportunities for future research.

2. BACKGROUND & RELATED WORK

An important aspect of explanations is the benefit they can bring to a system. Tintarev et al. define the following seven goals for explanations in recommender systems [8]: 1. *Transparency* to help users understand how the recommendations are generated and how the system works. 2. *Scrutability* to help users correct wrong assumptions made by the system. 3. *Trust* to increase users' confidence in the system. 4. *Persuasiveness* to convince users to try or buy items and enhance user acceptance of the system. 5. *Effectiveness* to help users make better decisions. 6. *Efficiency* to help users decide faster, which recommended item is the best for them and 7. *Satisfaction* to increase the user's satisfaction with the system. However, meeting all these criteria is unlikely, some of these aims are even contradicting such as persuasiveness and effectiveness. Thus, choosing which criteria to improve is a trade-off. Explanations might also differ by the degree of personalization. While non-personalized explanations use general information to indicate the relevance of a recommendation, personalized explanations clarify how

a user might relate to a recommended item [8].

Due to the benefits of explanations in mobile recommender systems, a lot of research has been conducted in this context. Since our work focuses on explanations aiming at improving transparency and scrutability in a recommender system, we investigated previous research in these two areas.

The work of Vig et al. [9] separates justification from transparency. While transparency should give an honest statement of how the recommendation set is generated and how the system works in general, justification can be refrained from the recommendation algorithm and explain why a recommendation was selected. Vig et al. developed a web-based **Tagsplanations** system where the recommendation is justified using relevance of tags. Their approach, as the authors noted, lacked the ability to let users override their inferred tag preferences.

Cramer et al. [3] applied transparent explanations in the web-based **CHIP** (Cultural Heritage Information Personalization) system that recommends artworks based on the individual user's ratings of artworks. The main goal of the work was to make the criteria more transparent the system uses to recommend artworks. It did so by showing the users the criteria on which the system based its recommendation. The authors argue that transparency increased the acceptance of the system.

An interesting approach to increase scrutability has been taken by Czarkowski [4]. The author developed **SASY**, a web-based holiday recommender system which has scrutinization tools that aim not only to enable users to understand what is going on in the system, but also to let them take control over recommendations by enabling them to modify data that is stored about them.

TasteWeights is a web-based social recommender system developed by Knijnenburg et al. [5] aiming at increasing inspectability and control. The system provides inspectability by displaying a graph of the user's items, friends and recommendations. The system allows control over recommendations by allowing users to adjust the weights of the items and friends they have. The authors evaluated the system with 267 participants. Their results showed that users appreciated the inspectability and control over recommendations. The control given via weighting of items and friends made the system more understandable. Finally, the authors concluded that such interactive control results in scrutability.

Wasinger et al. [10] apply scrutinization in a mobile restaurant recommender system named **Menu Mentor**. In this system, users can see the personalized score of a recommended restaurant and the details of how the system computed that score. However, users can change the recommendation behavior only by critiquing presented items via meal star ratings, no granular control over meal content is provided. A conducted user study showed that participants perceived enhanced personal control over given recommendations.

In summary, although previous research focused on increasing either scrutability or transparency in recommender systems, no research was conducted on how interactive explanations can increase transparency as well as scrutability in mobile recommender systems.

3. DESIGNING THE PROTOTYPE

Our system aims at offering shoppers a way to find nearby shopping locations with interesting clothing items while also supporting them in decision making by providing interactive explanations. Mobile recommender systems use a lot of situational information to generate recommendations, so it might not always be clear to the user how the recommendations are generated. Introducing transparency can help solving this problem. However, mobile devices require even more considerations in the design and development (e.g. due to the small display size). Thus, these should also be taken into account when generating transparent explanations. Moreover, the explanation framework should generate textual explanations that make it clear to the user how her preferences are modeled. In order not to bore the user, explanations must be concise and include variations in wording. Furthermore, introducing transparency alone might not be enough because users often want to feel in control of the recommendation process. The explanation goal scrutability addresses this issue by letting users correct system mistakes. There have been several approaches to incorporate scrutable explanations to traditional web-based recommender systems. However, more investigation is required in the area of mobile recommender systems. First of all, the system should highlight the areas of textual explanations that can be interacted with. Second, the system should allow the user to easily make changes and get new recommendations. While transparent and scrutable explanations are the main focus of this work, there are also some side goals, such as satisfaction and efficiency.

3.1 The Baseline

Shopr, a previously developed mobile recommender system serves as the baseline in our user study [6]. The system uses a conversation-based Active Learning strategy that involves users in ongoing sessions of recommendations by getting feedback on one of the items in each session. Thus, the system learns the user's preferences in the current context. An important point is that the system initially recommends very diverse items without asking its users to input their initial preferences. After a recommendation set is presented, the user is expected to give feedback on one of the items in the form of *like* or *dislike* over item features (e.g. price of the item or color) and can state which features she in particular *likes* or *dislikes*. In case the user submitted a positive feedback, using the *refine* algorithm shows more similar items. Otherwise, the system concludes a negative progress has been made and *refocuses* on another item region and shows more diverse items. The algorithm keeps the previously critiqued item in the new recommendation set in order to allow the user to further critique it for better recommendations. The explanation strategy used in this system is very simple. An explanation text is put on top of all items, which tries to convey the current profile of the user's preferences. It allows the user to observe the effect of her critiques and to compare the current profile against the actually displayed items. An example for such an explanation text is *"avoid grey, only female, preferably shirt/dress"*.

3.2 How Explicit Feedback Affects Weights

The modeling of the user's preferences is an important part of the proposed explanation generation strategy and feedback model and is adapted from the approach of *Shopr* [6], described in the *Baseline* section. It is modeled as a search query q with weights for values of features (e.g. *red* is a possible value of the feature *color*). For each feature,

there is a weight vector that allows the prioritization of one feature value over another. A query q for a user looking for only red dresses from open shops in 2000m reach would look like this (we here assume that each item has only the two features 'color' and 'type'):

$$q = ((distance \leq 2000m) \wedge (time\_open = now + 30min)), \{color_{red,blue,green}(1.0, 0, 0), type_{blouse,dress,trousers}(0, 1.0, 0)\} \quad (1)$$

Our system uses two types of user feedback. One of them is by critiquing the recommended items on their features (which was already provided in the baseline system, see *section 3.1*). The other is by correcting mistakes regarding the user's preferences via explicit preference statement. Explanations are designed to be interactive, so that the user can state her actual preference over feature values after tapping on the explanation. If the user states interests on some feature values, a new value vector will be initialized for the query with all interested values being assigned equal weight summing to 1.0 and the rest having 0.0 weight. That means that the system will focus on the stated feature values, whereas the other values will be avoided. For example if a user interacts with the explanation associated with the query presented in *equation 1* and states that she is actually only interested in blue and green, then the resulting new weight vector would look like the following (assuming that we only distinguish between three colors) which will influence the search query and thus the new recommendations:

$$feedback_{positive}(blue, green) : color_{red,blue,green}(0.0, 0.5, 0.5) \quad (2)$$

3.3 Generating Interactive Explanations

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. In order to explain the current state of the user model (which stores the user's preferences) and the reasoning behind recommendations, two types of explanations are defined: *recommendation-* and *preference* explanations.

3.3.1 Interactive Recommendation Explanations

Recommendation explanations are interactive textual explanations. Their first aim is to justify why an item in the recommendation set is relevant for the user. Second, they let the user make direct changes to her inferred preferences. The generation is based on the set of recommended items, the user model and the mobile context.

Argument Assessment.

Argument assessment is used to determine the quality of every possible argument about an item. The argument assessment method is based on the method described in [1] . It uses Multi-Criteria Decision Making Methods (MCDM) to assess items I on multiple decision dimensions D (e.g. features that an item can have) by means of utility functions. Dimensions in the context of this recommender system are features and contexts. The method described in [1] uses four scores, which lay a good foundation for the method in this work. However, their calculations have to be adapted to the underlying recommendation infrastructure to produce meaningful explanations.

Local score $LS_{I,D}$ measures the performance of a dimension without taking into account how much the user values that dimension. Our system uses feature value weight vectors to represent both item features and features in a query, which represents the current preferences of the user. Local score of a feature is the scalar product of the weight vector (for that feature) in the query with respective weight vector in the item's representation. It is formalized as below, where $w_{I,D}$ represents the feature value weight vector for item dimension D and $w_{Q,D}$ represents the feature value weight vector for query dimension D and n stands for the number of feature values for that dimension:

$$LS_{I,D} = \sum_{i=0}^{n-1} w_{I,D}(i).w_{Q,D}(i) \quad (3)$$

Explanation score $ES_{I,D}$ describes the explaining performance of a dimension. The weight for each dimension is calculated dynamically by using a function that decreases the effects of the number of feature values in each dimension. It is formalized as follows, where $length_{w_D}$ denotes the number of feature values in a specific dimension D and $length_{total\_attribute\_values}$ the total number of feature values for all dimensions. Using the square root produced good results since it limits the effect of number feature values on the calculation of weights.

$$w_D = \sqrt{\frac{length_{w_D}}{length_{total\_attribute\_values}}} \quad (4)$$

With the following dynamically calculated weight for a dimension, explanation score of the dimension can be calculated by multiplying it with the local score of that dimension:

$$ES_{I,D} = LS_{I,D}.w_D \quad (5)$$

Information score IS_D measures the amount of information provided by a dimension. The calculation of information score suggested by [1] is preserved as it already lays a good foundation to reason whether explaining an item from a given dimension provides a good value. So, it can be defined as follows where R denotes the range of explanation scores for that dimension for all recommended items and I denotes the information that dimension provides for an item:

$$IS_D = \frac{R + I}{2} \quad (6)$$

Range R is calculated as the difference between the maximum and minimum explanation score for the given dimension for all recommended items, namely $R = max(ES_{I,D}) - min(ES_{I,D})$. Information I, however, is calculated quite differently from the strategy proposed by [1]. In their system, a dimension provides less and less information as the number of items to be explained from the same dimension increases. This does not apply to the context of the clothing recommender developed for this work. An item could still provide good information if not there are not so many items that can be explained from the same feature value. For instance, it is still informative to explain an item from the color blue; although another item is also explained by the same dimension (color) but from a different value, let's say green. Therefore, I is calculated as a function of the size

of recommendation set (n) and number of items in the set that has the same value for a dimension (h): $I = \frac{n-h}{n-1}$.

Global score GS_I measures the overall quality of an item in all dimensions. It is the mean of explanation scores of all of its dimensions. The following formula demonstrates how it is formalized, where n denotes the total number of all dimensions and ES_{I,D_i} the explanation score of an item on i_{th} dimension.

$$GS_I = \frac{\sum_{i=0}^{n-1} ES_{I,D_i}}{n} \quad (7)$$

The above-defined methods for calculating explanation and information scores are only valid for item features. Explanations should also include relevant context arguments. In order to support that, every context instance that is captured and used by the system in the computation of the recommendation set should also be assessed. The explanation score of a context dimension is calculated using domain knowledge. The most important values for the context gets the highest explanation score and it becomes lower and lower as the relevance of the value of the context decreases. For example, for location context, the explanation score is inversely proportional to the distance between the current location of the user and the shop where the explained item is sold. Explanation score gets higher as the distance gets lower. Information score is calculated with the same formula defined earlier for features $IS_D = \frac{R+I}{2}$, but Information I slightly changes. As proposed earlier, it is calculated using the formula $I = \frac{n-h}{n-1}$, but in this case h stands for the number of items with similar explanation score.

Argument Types.

In order to generate explanations with convincing arguments, different argument aspects are defined by following the guidelines for evaluative arguments described in [2]. Moreover, the types of arguments described in [1] are taken as a basis. First of all, arguments can be either *positive* or *negative*. While positive arguments are used to convince the user to the relevance of recommendations, negative arguments are computed so that the system can give an honest statement about the quality of the recommended item. The second aspect of arguments is the type of dimension they explain, *feature* or *context*. Lastly, they can be *primary* or *supporting* arguments. Primary arguments alone are used to generate concise explanations. Combination of primary and supporting arguments are used to generate detailed explanations. We distinguish between five argument types: *Strong primary feature arguments*, *Weak primary feature arguments*, *Supporting feature arguments*, *Context arguments* and *Negative arguments*.

Explanation Process.

The explanation process is based on the approach described in [1] but it is adapted to use the previously defined argument types. Different from the system in [1], explanations are designed to contain multiple positive arguments on features. Negative arguments are generated but only displayed when necessary by using a ramping strategy. *Figure 1* shows the process to select arguments. It follows the framework for explanation generation described in [2]

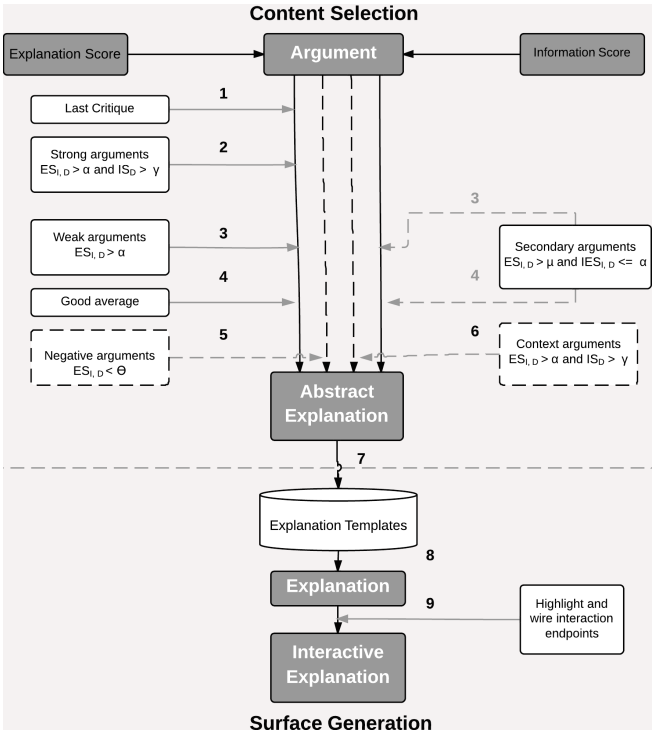


Figure 1: Generation of explanations.

as the process is divided into the selection and organization of explanation content and the transformation in a human readable form.

Content Selection. The argumentation strategy selects arguments for every item I separately. One or more primary arguments are selected first to help the user to instantly recognize why the item is relevant. There are four alternative ways to select the primary arguments (alternatives 1 to 4 in *figure 1*). First alternative is that the item is in the recommendation set because it was the last critique and it was carried (1). Another is that the system has enough *strong arguments* to explain an item (2). If there are not any strong arguments, the strategy checks if there are any *weak arguments* (3). In case there are one or more weak arguments, the system also adds supporting arguments to make the explanation more convincing. Finally, if there are no weak arguments too, then the item is checked if it is a good average by comparing its global score GS_I to threshold β (4). If so, similar to alternative (3), supporting arguments are also added to increase the competence of the explanation. Otherwise the strategy supposes that the recommended item is serendipitous and added to the set to explore the user's preferences. With one or more primary arguments, the system checks if there are any negative arguments and context arguments to add (5 and 6).

Surface Generation. The result of the content selection is an *abstract explanation*, which needs to be resolved to something the user understands. This is done in the surface generation phase. Various explanation sentence templates are decorated with either feature values or context values (7 and 8). Explanation templates are sentences with placeholders for feature and context values stored in XML format. The previously determined primary argument type

Table 1: Text templates for recommendation explanations.

Text template	Example phrase
Strong argument	*"Mainly because you currently like X."*
Weak argument	*"Partially as you are currently interested in X."*
Supporting argument	*"Also, slightly because of your current interest in X."*
Location context	*"And it is just Y meters away from you."*
Average item	*"An average item but might be interesting for you."*
Last critique	*"Kept so that you can keep track of your critiques."*
Serendipity	*"This might help us discovering your preferences."* or *"A serendipitous item that you perhaps like."*
Negative argument	*"However, it has the following feature(s) you don't like: X, Y [...]."*

is used to determine which type of explanation template to use. Feature values in the generated textual output are then highlighted and their interaction endpoints are defined (9). The resulting output is a textual explanation, highlighted in the parts where feature values are mentioned. They are interactive such that, after the user taps on the highlighted areas, she can specify what she exactly wants.

3.3.2 Interactive Preference Explanations

Preference explanations have got two main goals. First, they aim to let the user inspect the current state of the system's understanding of the user's preferences. Second, they intend to let the user make direct changes to the preference. Two main types of preferences explanations are defined, *interactive textual explanations* and *interactive visual explanations*.

Generating Textual Preference Explanations.

The only input to textual preference explanation generation algorithm is the user model. For each dimension D the algorithm can generate interactive explanations. Dimensions are features that an item can have. The algorithm distinguishes between four feature value weight vectors, indicating different user preferences: First, the user is indifferent to any feature value. Second, the user is only interested in a set of feature values. Third, the user is avoiding a set of feature values. And fourth, the user prefers a set of feature values over others.

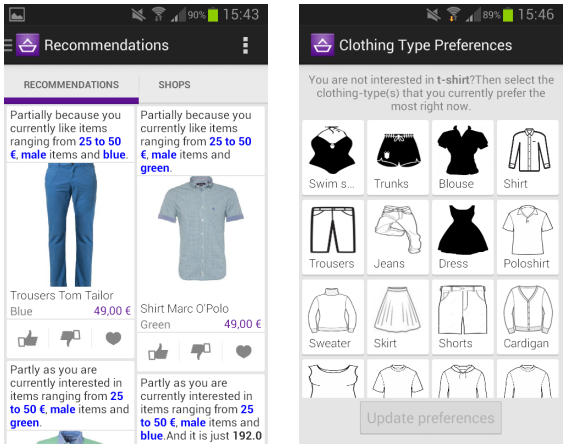
Generating Visual Preference Explanations.

Visual preference explanations are generated also by using the user model, more specifically by making use of the array of feature value weight vectors, which represents the user's current preferences. For each feature, there is already a feature value weight vector, which indicates the priorities of the user among feature values. All those weights are between 0.0 and 1.0 summing up to 1.0. They could be scaled to a percentage to generate charts showing the distribution of percentage of interests for feature values.

In order to generate charts, it is also required to determine with which color and description a feature value will be represented in a chart. In order to support that, a feature value

Table 2: Text templates for preference explanations.

Text template	Example phrase
Only some values	*"You are currently interested only in X, Y [...]."* The word "only" in the text is emphasized in bold.
Avoiding some values	*"You are currently avoiding X, Y [...]."* The word "avoiding" is emphasized in bold.
Preferably some values	*"It seems, you currently prefer X, Y [...]."*
Indifferent to feature	*"You are currently indifferent to X feature".*



(a) (b)

Figure 2: Recommendation list (a) and explicit preference feedback screen (b).

appearing in the chart is modeled with its weights (scaled to a percentage), color and description in the user interface. *Figure 5* illustrates this chart representation.

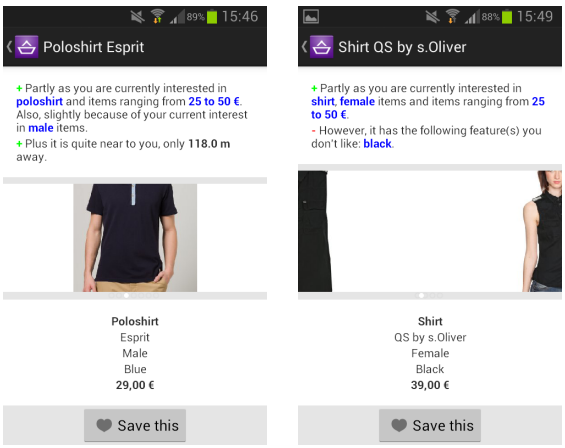
3.3.3 Using Text Templates Supporting Variation

XML templates are used to generate explanation sentences for the different user preference types in English language. Those templates contain placeholders for feature and context values which are replaced during the explanation generation process. For *recommendation explanations*, there are a few sentence variations for almost every type of arguments. See *table 1* for examples of the different text templates for recommendation explanations. These templates can be used in combination with each other. For example, supporting arguments can support a weak argument. In such cases, argument sentences are connected using conjunctions.

Similar mechanism is also used for the *preference explanations*. However, to keep it simple, variation is not provided, as the number of features to explain is already limited. See *table 2* for selected examples of several text templates for preference explanations.

3.4 Interaction and Interface Design

The first issue was to clarify how to integrate the interaction process with textual explanations. It was envisioned to give the user the opportunity to tap on the highlighted



(a) (b)

Figure 3: Detailed information screens of items.

areas of the explanation text to state her actual preferences on a feature. This leads to a two-step process. First, the user sees an item with an explanation including highlighted words (highlighted words are always associated with a feature, see *figure 2a*) and taps on one of them (e.g. in *figure 2b*, "t-shirt" was tapped). Then the system directs the user to the screen where the user can make changes. In this second step, she specifies which feature values she is currently interested in. Lastly, the system updates the list of recommendations which complets a recommendation cycle. Note that the critiquing process and associated screens from the project *Shopr*, which is taken as a basis (see *section 3.1*) are kept in the developed system. Eventually, the interaction is a hybrid of critiquing and explicitly stating current preferences. On top of each explicit feedback screen, a text description of what is expected from the user is given.

Due to the applied ramping strategy mentioned in *section 3.3.1*, all extra arguments in explanations that are not important were not shown as explanations in the list of recommendations but in the screen where item details are presented. Tapping on an item picture accesses that screen. Here, the user can also browse through several pictures of an item by swiping the current picture from right to left (see *figure 3b*). In order to make it obvious for the user, the sentences with positive arguments always start with a green "+" sign. Negative arguments sentences, on the other hand, always start with a red "-" sign (see *figure 3*).

The next issue was to implement preference explanations, what we call *Mindmap feature*. Mindmap feature is the way that system explains its mental map about the preferences of the user. The overview screen for mindmap was designed to quickly show the system's assumptions about the user's current preferences. To keep it simple but yet usable, only textual explanations are used for each feature (see *figure 4b*). In order to make it easy for the user to notice what is important, the feature values used in the explanation text are highlighted. Moreover, every element representing a feature is made interactive. This lets the user access the explicit feedback screen to provide her actual preferences.

The user should also be able to get more detailed visual information for all the features. In order to achieve that, a different "drill down" screen for all screens was developed as part of the mindmap feature. *Figure 5* shows the mindmap detail screens for the clothing color feature. The user's preferences on feature values are represented as a chart. Every feature value is displayed as a different color in the charts. One of the most important features is that the highlighted parts of the explanation texts and the charts are interactive as well which lets the user access the explicit feedback screen to provide her actual preferences.

The full source code and resources for the Android app and the algorithm are available online[1].

4. USER STUDY

The main three goals of the evaluation are: First, to find out whether transparency and user control can be improved by feature-based personalized explanations supported by scrutable interfaces in recommender systems. Second, to find out whether side goals such as higher satisfaction are achieved and lastly to see whether other important system goals such as efficiency are not damaged.

4.1 Setup

The *test hardware* is a 4.3 inch 480 x 800 resolution Android smartphone (Samsung Galaxy S2) running the *Jelly Bean* version of the Android operating system (4.1.2).

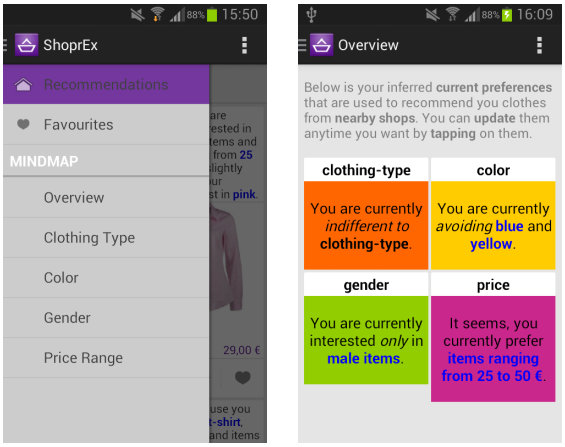
Two *variants* of the system are put to the test. In order to refrain from the effects of different recommender algorithms, both variants use the same recommendation algorithm which uses diversity-based Active Learning [6]. Moreover, critiquing and item details interfaces are exactly the same. The difference lies in the explanations: The *EXP* variant refers to the proposed system, described in the previous section. In order to test the value of the developed explanations and scrutinization tools, a baseline (*BASE* variant) to compare against is needed (see *subsection 3.1*). The study is designed as within-subject to keep the number of testers at a reasonable size. Thus one group of people tests both variants. Which system is tested first was flipped in between subjects so that a bias because of learning effects could be reduced.

In order to create a realistic setup, it is necessary to generate a *data set* that represents real-world items. For that purpose, we developed a data set creation tool as an open-source project[2]. The tool crawls clothing items from a well-known online clothing retailer website. To keep the amount of work reasonable, items were associated with an id, one of 19 types of clothing, one of 18 colors, one of 5 brands, the price (in Euro), the gender (male, female or unisex) and a list of image links for the item. The resulting set is 2318 items large, with 1141 for the male and 1177 for the female gender.

For the study, *participants* of various age, educational background and current profession were looked for. Overall 30 people participated, whereas 33% of users were female and 67% were male.

The actual *testing procedure* used in the evaluation was structured as follows: We first asked the participants to provide background information about themselves, such as demographic information and their knowledge about mobile systems and recommender systems. Next, the idea of the

[1] `https://github.com/adiguzel/Shopr`
[2] `https://github.com/adiguzel/pickpocket`



(a) (b)

Figure 4: Navigation Drawer (a) and Overview (b).

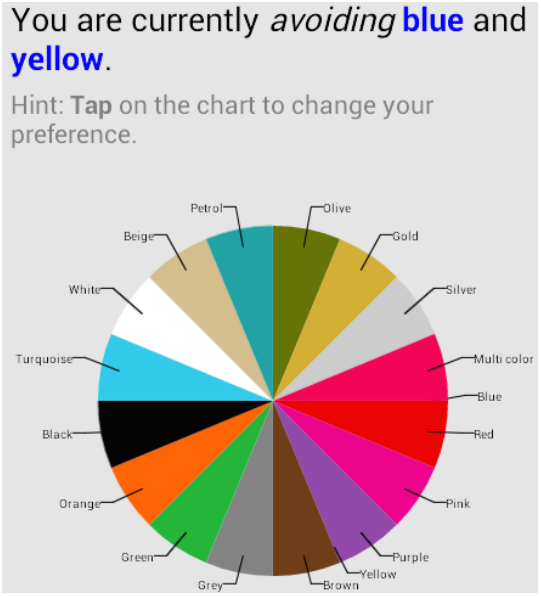


Figure 5: Mindmap detail screens for color.

system was introduced and the purpose of the user study was made clear. We chose a realistic scenario instead of asking users to find an item they could like:

Task: *Imagine you want to buy yourself new clothes for an event in a summer evening. You believe that following type of clothes would be appropriate for this event: shirt, t-shirt, polo shirt, dress, blouse or top. As per color you consider shades of blue, green, white, black and red. You have a budget of up to € 100. You use the Shopr app to look for a product you might want to purchase.*

After introducing them to the task, users were given hands on time to familiarize themselves with the user interface and grasp how the app works. After selecting and confirming the choice for a product, the task was completed. Then testers were asked 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) and offer any general feedback and observations. After having tested both variants, participants stated which variant they preferred and why that was the case.

4.2 Results

The testing framework applied in the user study is a subset of the aspects that are relevant for critiquing recommenders and explanations in critiquing recommenders. It follows the user-centric approach presented in [7]. The measured data is divided into four areas: transparency, user control, efficiency and satisfaction.

The means of the measured values for the most important metrics of the two systems, BASE denoting the variant using only simple non-interactive explanations, EXP the version with interactive explanations, are shown in *table 3*. Next to the mean the standard deviation is shown, the last column denoting the p-value of a one-tail paired t-test with 29 degrees of freedom (30 participants - 1).

In order to measure actual understanding after using a variant, users were asked to describe how the underlying recommendation system of that variant works. In general, almost all of the participants could explain for both recommenders that the systems builds a model of the user's preferences in each cycle and uses it to generate recommendations that can be interesting for the user.

On average, when asked if a tester understands the system's reasoning behind its recommendations, EXP performs better than BASE (mean average of 4.63 compared to 4.3 out of a 1-5 Likert scale). Further analysis suggests that the variant with interactive explanations (EXP) is perceived significantly more transparent than the variant with baseline explanations (one-tail t-test, $p<0.05$ with $p=0.018$).

Users were asked about the ease of telling the system what they want in order to measure the overall user control they perceived. Average rating of participants was better with EXP (4.33 versus 3.23). In a further analysis, EXP seemed significantly better in terms of perceived overall control than BASE (one-tail t-test, $p<0.05$ with $p=0.0003$).

When asked about the ease of correcting system mistakes, EXP performs a lot better than BASE (mean average of 4.36 compared to 3 out of a 1-5 Likert scale). Further analysis reveals that EXP is significantly better in terms of perceived scrutability than BASE (one-tail t-test, $p<0.05$ with $p=0.6.08E-06$).

Participants completed their task in average one cycle less using EXP than BASE (6.5 with EXP, 7.46 with BASE). However, one-tail t-test shows that EXP is not significantly better than BASE ($p>0.05$ with $p=0.14$).

The next part of measuring objective effort is done via tracking the time it took for each participant from seeing the initial set of recommendations until the target item was selected. On average BASE seems to be better with a mean session length of 160 seconds against 165 seconds. However, it was found not to be significantly more time efficient (one-tail t-test, $p>0.05$ with $p=0.39$). One reason for this could be that although EXP gives its users tools to update preferences over several features quickly, it has more detailed explanations. Thus, users spent more time with reading.

Users were asked about the ease of finding information and the effort required to use the system in order to get an idea about the system's efficiency. The participants' av-

Table 3: The means of some important measured values comparing both variations of the system.

	BASE mean	stdev	EXP mean	stdev	p value
Perceived transparency	4.3	0.70	4.63	0.49	**0.018**
Perceived overall control	3.23	1.04	4.33	0.71	**0.0003**
Scrutability	3	1.31	4.36	0.85	**6.08E-06**
Cycles	7.46	3.64	6.5	3.28	0.14
Time consumption	160 s	74	165 s	83	0.39
Perceived efficiency	3.43	1.13	4.33	0.75	**0.0003**
Satisfaction	3.76	0.85	4.43	0.56	**0.0004**

erage rating was better with EXP with 4.33 against 3.43 with BASE. Further analysis revealed that users perceived EXP significantly more efficient than BASE (one-tail t-test, $p<0.05$ with $p=0.0003$).

When inquired how satisfied participants were with the system overall, EXP performs better with 4.43 against 3.76. One-tail t-test suggests that this is a significant result ($p<0.05$ with $p=0.0004$).

Finally, participants were asked to pick a favorite from the two evaluated variants. 90% preferred the variant with interactive explanations (EXP) over the variant with simple non-interactive explanations (BASE), mostly because of the increased perception of control over recommendations.

5. CONCLUSION AND FUTURE WORK

This work investigated the development and impact of a concept featuring interactive explanations for Active Learning critique-based mobile recommender systems in the fashion domain. The developed concept proposes the generation of explanations to make the system more transparent while also using them as an enabler for user control in the recommendation process. Furthermore, the concept defines the user feedback as a hybrid of critiquing and explicit statements of current interests. A method is developed to generate explanations based on a content-based recommendation approach. The explanations are always made interactive to give the user a chance to correct possible system mistakes. In order to measure the applicability of the concept, a mobile Android app using the proposed concept and the explanation generation algorithm was developed. Several aspects regarding display and interaction design of explanations in mobile recommender systems are discussed and solutions to the problems faced during the development process are summarized. The prototype was evaluated in a study with 30 real users. The proposed concept performed significantly better compared to the approach with non-interactive simple explanations in terms of our main goals to increase transparency and scrutability and side goals to increasing perceived efficiency and satisfaction. Overall, the developed interactive explanations approach demonstrated the user appreciation of transparency and control over the recommendation process in a conversation-based Active Learning mobile recommender system tailored to a modern smartphone platform. Some changes, such as increasing the number of recommendations, skipping to the next list of recommendations without critiquing and having more item attributes for critiquing, could make the application even more appealing.

Future development may also include the creation of more complex recommendation scenarios to test the capability of the proposed concept even further. One can add more item features to critique and also take the user's mobile context (e.g. mood and seasonal conditions) into account during the recommendation process. Furthermore, future research might study the generation of interactive explanations for systems with rather complex recommendation algorithms. Interactive explanations might make adjustable parts of the algorithm transparent and allow the user to change them.

6. REFERENCES

[1] R. Bader, W. Woerndl, A. Karitnig, and G. Leitner. Designing an explanation interface for proactive recommendations in automotive scenarios. In *Proceedings of the 19th International Conference on Advances in User Modeling*, UMAP'11, pages 92–104, Berlin, Heidelberg, 2012. Springer-Verlag.

[2] G. Carenini and J. D. Moore. Generating and evaluating evaluative arguments. *Artif. Intell.*, 170(11):925–952, Aug. 2006.

[3] H. Cramer, V. Evers, S. Ramlal, M. Someren, L. Rutledge, N. Stash, L. Aroyo, and B. Wielinga. The effects of transparency on trust in and acceptance of a content-based art recommender. *User Modeling and User-Adapted Interaction*, 18(5):455–496, Nov. 2008.

[4] M. Czarkowski. *A Scrutable Adaptive Hypertext*. PhD thesis, University of Sydney, 2006.

[5] B. P. Knijnenburg, S. Bostandjiev, J. O'Donovan, and A. Kobsa. Inspectability and control in social recommenders. In *Proceedings of the Sixth ACM Conference on Recommender Systems*, RecSys '12, pages 43–50, New York, NY, USA, 2012. ACM.

[6] B. Lamche, U. Trottman, and W. Wörndl. Active learning strategies for exploratory mobile recommender systems. In *Proceedings of CaRR workshop, 36th European Conference on Information Retrieval*, Amsterdam, Netherlands, Apr 2014.

[7] P. Pu, L. Chen, and R. Hu. A user-centric evaluation framework for recommender systems. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11, pages 157–164, New York, NY, USA, 2011. ACM.

[8] N. Tintarev and J. Masthoff. Evaluating the effectiveness of explanations for recommender systems. *User Modeling and User-Adapted Interaction*, 22(4-5):399–439, Oct. 2012.

[9] J. Vig, S. Sen, and J. Riedl. Tagsplanations: Explaining recommendations using tags. In *Proceedings of the 14th International Conference on Intelligent User Interfaces*, IUI '09, pages 47–56, New York, NY, USA, 2009. ACM.

[10] R. Wasinger, J. Wallbank, L. Pizzato, J. Kay, B. Kummerfeld, M. Böhmer, and A. Krüger. Scrutable user models and personalised item recommendation in mobile lifestyle applications. In *User Modeling, Adaptation, and Personalization*, volume 7899 of *Lecture Notes in Computer Science*, pages 77–88. Springer Berlin Heidelberg, 2013.