Predicting Answers in Preference Elicitation Dialogs

Ralph Samer and Martin Stettinger and Alexander Felfernig and Muesluem Atas[1] and Robert Zupan and Manuel Henrich[2]

Abstract. The elicitation of user preferences represents an effective means to identify the relevant user preference requirements for a configuration task. One common method to determine the preferences of users is to survey users using dialogs that consist of (multiple-choice) questions with selectable answers. A major drawback of dialog-based preference elicitation is that many new users are not willing to answer a fairly large number of questions which is a prerequisite for the identification of user constraints. To that end, we have developed a novel similarity-based approach which aims to solve such ramp-up (cold-start) scenarios by automatically completing a set of remaining questions in a preference elicitation dialog given a small set of pre-answered questions. Our approach has been evaluated with two small real-world datasets. Initial evaluation results reveal that our approach is able to find the most probable answers a respondent is likely to give to a set of remaining questions. First insights of an evaluation also show that our approach can keep the number of initial questions at a very low level, meaning that only between 35% and 50% of questions have to be asked in most cases in order to predict the complete set of user requirements. The results also indicate that this level can be further reduced with increasing amounts of training data.

1 Introduction

Configuration systems are tools that help users to find matching solutions / objects (e.g., items or services) in large object spaces or to generate relevant solutions which satisfy all pre-defined constraints [1, 8, 13, 14]. The set of constraints usually represents the combination of a user's preferences (*user constraints*) and the given knowledge base (*background knowledge constraints*). In this context, the active querying of user preferences within the scope of online sessions [5] represents a proper method to elicit the preferences of a user in an interactive and iterative fashion.

The identification of matching solutions often presupposes an extensive description of a user's preferences as a major precondition to gather a representative and complete set of user constraints. It is important to mention that the complete set of user constraints must meticulously reflect the imprint of the individual needs the user has. This requires diligent questioning of the user's individual preferences in the form of *preference elicitation dialogs*. These dialogs typically consist of a number of multiple-choice questions specially geared to the respective configuration domain and task. The main purpose of preference elicitation dialogs is to collect the user preferences in order to create all relevant user constraints for the configuration environment.

[1] Graz University of Technology, Austria, email: {rsamer, martin.stettinger, alexander.felfernig, muesluem.atas}@ist.tugraz.at
[2] UNiQUARE Software Development GmbH, Austria, email: {robert_zupan, manuel.henrich}@uniquare.com

A common challenge that often arises in this context is the time-intensive preference elicitation process which involves many questions and, hence, lengthy preference elicitation dialogs. Existing research [10, 11, 12] shows that the explicit querying of all user preferences is known to be very challenging which can result in situations where users tend to leave the active querying process and discontinue the dialog before all relevant user preferences can be captured.

Therefore, smart mechanisms are needed to facilitate the elicitation process of user preferences. To that end, we devised a novel approach to reduce the length of preference elicitation dialogs. A major advantage of our approach is to address cold-start issues which occur whenever a configuration system has to capture the preferences of a new user in a dialog-based preference elicitation process. Preliminary evaluation results indicate that the developed approach is able to effectively reduce lengthy preference elicitation dialogs which typically consist of more than 20 multiple-choice questions. According to our results, our approach achieves a reduction of up to 65% of questions for most users (i.e., only 35% of all questions need to be answered by these users in order to predict the missing answers and to derive the complete set of user constraints).

The remainder of this paper is structured as follows. In Section 2, we introduce and explain our similarity-based prediction approach to support new users in preference elicitation dialogs. Section 3 focuses on the evaluation of our approach and a brief discussion of the evaluation results. In Section 4, we give an overview of existing research that is related to our work. Finally, Section 5 concludes this paper and gives an outlook on ideas regarding future work.

2 Methodology

In this section, we discuss an approach which predicts the (most probable) answers of a user to the remaining questions in the context of multiple-choice preference elicitation dialogs. The purpose of preference elicitation dialogs is to support the collection of user preferences necessary to define user constraints for a configuration system.

2.1 Preliminaries

In order to better understand the approach discussed in this paper, we first introduce configuration systems and provide some formal definitions. In general, a configuration system represents a sophisticated search-based tool that finds solutions for problems by taking into account a variety of variables and constraints. A configuration problem can be regarded as a *constraint satisfaction problem* which exploits search heuristics (i.e., search strategies) to find or generate relevant solutions. In this particular context, a constraint satisfaction problem defines a task (V, C) where V is a set of variables and C represents a set of constraints. An assignment of a variable in V can be

regarded as consistent for the task (V, C) if it satisfies all constraints in C (i.e., does not violate any of the constraints in C). The major objective of a constraint solver is to ensure that all variable assignments are consistent with C such that suitable solutions can be found and presented – in case variable assignments are inconsistent with C, diagnose and repair mechanisms can be applied (for further details, we refer to [7, 8]).

The set of constraints C typically includes all system constraints ($SC \subset C$) as well as all user constraints ($UC \subset C$). The system constraints (SC) reflect the relevant background knowledge and the user constraints (UC) represent unary constraints which refer to concrete customer requirements. The system constraints have to be predefined – they are typically specified by the system engineers during the construction of the system. However, the user constraints need to be collected by the users / customers. One common way that is often used in practical scenarios is to precisely ask a customer a number of questions in a dialog-based session. In these dialogs, many questions are asked to elicit the user preferences and to derive the user constraints (UC) for the underlying constraint satisfaction problem (V, C). This usually represents a high effort for most users and often results in undesired outcomes where users often leave the dialog session before having answered all questions and seen any solution presented by the configuration system. However, in order to determine the relevant user constraints for a new user, the preferences of the user have to be captured through a dialog-based preference elicitation process. This situation is referred to as the *cold-start problem* and constitutes a common well-known problem in the context of recommendation and configuration systems. In the remainder of this section, we describe a new approach to address this cold-start problem for configuration systems.

2.2 Technical Approach

In order to reduce the overall effort for a user in preference elicitation dialogs, we introduce a new approach to facilitate survey-based preference elicitation. Our approach supports preference elicitation dialogs with multiple-choice questions. It utilizes a neighborhood-based search strategy to determine the most suitable multiple-choice questions which are needed to guide the user through a shortened dialog and to infer remaining preferences from historic dialogs of like-minded users. This shortened dialog allows to determine the preferences of a user based on the answers declared by the user. The set of preferences can then be completed by inferring missing preferences from historical dialogs of other like-minded users. The answers of the remaining questions given by the like-minded users define the basis to complete the set of a user's preferences.

In order to make suitable predictions of missing answers in a dialog session, the answering behavior of the respondent needs to be captured. This requires our approach to pursue two major steps which are (1) determine an initial set of questions which are most appropriate to capture a respondent's answering behavior in an accurate fashion, and (2) collect the answers to these questions given by the active respondent. In the first step, our approach seeks to minimalize the number of initial questions, whereas in the second phase our approach strives to recognize the answering behavior of the active respondent.

Table 1 depicts an example setting of 3 users who have already answered all relevant questions. Each row corresponds to the answer results of a user given in a preference elicitation dialog. Each column refers to a multiple-choice question and presents the answer-combinations (i.e., choice combinations) given by the users / respondents. For example, the value "01" in Table 1 expresses that the user has selected the second answer but not the first answer. We use this table as a small working example to briefly explain the technical details of our approach in the remainder of this section.

	q_1	q_2	q_3	q_4
u_1	01	11	110	1010
u_2	10	11	101	1000
u_3	11	11	110	1010
u_a	?	?	110	1010

Table 1. Basic example to demonstrate the process of multiple-choice preference elicitation dialogs. All multiple-choice questions $\{q_1, q_2, q_3, q_4\}$ have been answered by different users $\{u_1, u_2, u_3\}$ in previous dialog sessions. The respondents' answers are encoded as binary numbers and indicate which answer-fields (choice-fields) of the question have been selected by the respondents.

In our approach, the initial questions represent those questions for which all previous respondents have provided the most diverging answers (i.e., strong variety of multiple-choice answer-combinations). These questions represent the best entry point in a preference elicitation dialog to be presented to a new respondent. The initial questions are determined based on the answers given to these questions in previous sessions by other respondents. For each question, we first determine the most common / frequent multiple-choice answer and then count how many respondents have provided an answer that differs from the most frequent answer.

Since each question can have a different number of multiple-choice options (e.g., 2, 3, or 4 answer-fields can be selected), a special peculiarity needs to be considered to determine the initial questions. Regardless of the respondents' answering-behavior, the answers of questions with less choice-options (e.g., only 2 answer-fields) typically tend to diverge more than the answers of questions with many choice options (e.g., 4 answer-fields). For this reason, we multiply our counted number (see above) with the binary logarithm of the choice options the question has. This simple mathematical operation is shown in Formula 1 where q_x refers to a specific question, *counted_value* represents the number answers that differ from the majority answer of q_x, and *answer_options* corresponds to the number of choices that can be selected for q_x. For example, the most frequent answer of question 3 presented in Table 1 is 110 and, hence, the divergence of question q_3 can be estimated by applying Formula 1 ($div(q_3) = \frac{1}{ld(3)} = 0.63$). The divergences of the remaining questions are $div(q_1) = \frac{3}{ld(2)} = 3.0$, $div(q_2) = \frac{0}{ld(2)} = 0.0$, $div(q_4) = \frac{1}{ld(4)} = 0.5$. Assuming that a new user u_a starts a dialog, we would thus choose q_3 and q_4 as initial questions (due to their high divergence) to determine the first two ($j = 2$) initial questions.

$$div(q_x) = \frac{counted_value}{ld(answer_options(q_x))} \tag{1}$$

Based on the answers a respondent gives to the initial questions, a neighborhood-based mechanism attempts to find k (neighbor) respondents in the second step, who have provided answers that are either identical or closely similar (i.e., below a *pre-defined threshold* parameter α) to the answers given by the current respondent. The k neighbor respondents are determined by making pairwise comparisons between the active respondent and the other users. We use the *cosine similarity* to estimate how closely related the active respondent is to another respondent by taking into account their individual answers of the initial questions. For example, if we want to find the

$k = 2$ nearest neighbors of respondent u_a, we need to investigate the answers from Table 1 which were given to our initial questions q_3 and q_4. As can be seen in the table, the most similar users of u_a are u_3 and u_1. These closely similar users can be regarded as the top two nearest neighbors or u_a. In addition to the pairwise comparison and the calculation of the cosine similarities, we also want to ensure that the algorithm only takes into account closely similar nearest neighbors. For this reason, we use a pre-defined parameter (denoted as α) which defines a threshold for the minimum similarity that must exist between the active respondent u_a and another respondent u_x such that u_x can be considered as nearest neighbor.

In case (at least or exactly) k nearest-neighbor respondents can be identified, the answers to the dialog's remaining questions of these neighbors can be compared. If these answers do not strongly diverge (in terms of the average *hamming distance*[3]), a consensus-based aggregation function (majority of the answers) is applied to identify the most probable answers the active respondent is likely to give to the remaining questions and the session can be closed without asking any further questions. In case no or less than k neighbors could be found or the answers of the neighbors diverge significantly, a new iteration starts and the whole process is repeated. Our approach then attempts to find the next best question for the current respondent (i.e., the next most diverging question). After the respondent has answered the next question, the k nearest neighbors are determined once again. The whole process is repeated until k nearest neighbors are found for whom the remaining questions have been answered in a quite similar fashion (the answers do not differ so much from each other and the standard deviation of the differences lies below the aforementioned threshold).

Due to the strict nature of selecting k nearest neighbors and our threshold parameter α, there exist some rare cases where our approach always fails to predict answers for new respondents even if they have already answered almost all questions. The number of respondents for whom this happens can be considered as quite low (see Section 3) and mostly depends on the strictness of finding nearest neighbors which can be controlled via the parameters α and k.

3 Evaluation

In this section, we present some preliminary evaluation results of our approach (see Section 2). In order to carry out an early evaluation, we used two real-world datasets to estimate the prediction quality of our approach. The main purpose of the first dataset is to investigate the scenario in which there are fewer questions overall, but enough historical data is available to train our algorithm. Using the second dataset, we want to evaluate our algorithm for certain situations in which exactly the opposite is the case (i.e., fewer transactions and longer dialogs with more questions). The first dataset (denoted as *DS-A*) consists of 20 multiple-choice questions, 86 answers / options (i.e., questions typically have 4.3 possible answers on average), and 125 historic sessions of different users. In this dataset the number of user sessions outweighs the number of questions. In contrast to that, the second dataset (denoted as *DS-B*) is more sparse in terms of the number of historic user sessions but contains a significantly higher number of questions. It consists of 78 multiple-choice questions, 301 answers / options (i.e., questions typically have 3.85 possible answers on average), and only 32 historic sessions of different users.

[3] The decision criteria was that the average hamming distance of the neighbor's answers given to all remaining questions must be below a pre-defined threshold. The threshold value is learned via grid-search (see evaluation in Section 3).

To evaluate our approach, we split each dataset into a training set (80%) and a test set (20%) and applied grid-search as well as *k-fold cross validation* (k=10) for the purpose of selecting the best hyperparameters (see parameters mentioned in Section 2). Furthermore, we compared the performance results of our approach with a basic neighborhood-based collaborative filtering approach which represents our evaluation baseline. The prediction quality of our approach was measured in terms of accuracy and the prediction error (mean and standard deviation). Each training and test sample referred to a single dialog session of a different user. During the training phase, we took into account the entire content of each training sample. To test and evaluate our approach and baseline approach, we only presented the answers the users from the test set have provided for the initial questions (the *pre-answered questions*) to the algorithm in order to predict the answers of the remaining questions. These predicted answers were then compared with the actual answers these users from the test set have given for the remaining questions, in order to compute the prediction accuracy (see below).

A major difference that exists between our approach and the baseline approach is the chronological order of the questions. While the baseline approach (basic collaborative filtering) allows users to answer questions in an arbitrary order before predictions are made in a dialog session, the approach presented in this paper predetermines the sequence of the initial questions that have to be answered by the respondent (see description in Section 2). The main reason for choosing such a baseline is that our approach and this baseline approach are able to predict multiple answers per question. This stands in stark contrast to existing approaches which are mentioned in Section 4. Moreover, existing preference elicitation approaches do not support multiple-choice responses and are thus inappropriate for drawing reasonable comparisons between these approaches and our approach (see Section 4 for further details).

The two tables Table 2 as well as Table 3 show a comparison of the evaluation results between our approach and the baseline after each iteration for dataset *DS-A* and *DS-B*, respectively. In these tables, the column *pre-answered questions* indicates how many pre-answered questions of a user (from the test set) were provided as input to the approach (or baseline approach, respectively) in order to predict all answers of the remaining questions. Both tables present the average prediction accuracy achieved for all users from the test set. Thereby, we once measured the average accuracy in terms of the correctness of the predictions from the viewpoint of the whole answer (see column *Avg. Accuracy (answer combinations)*). This means that in this case partly-correct answers are not considered as correct and only a completely correct answer is regarded as a correct answer (i.e., all multiple-choice options must be correct). Moreover, we also compared the accuracy results between our approach and the baseline approach from the perspective of individual answers (see column *Avg. Accuracy (individual answers)*). In this case, the accuracy is measured on the level of individual answers (i.e., selected multiple-choice options / answers of a question) given for a question which means that partly-correct answers are considered as partly-correct rather than completely wrong. The *user coverage* rates are also stated in a separate column in both tables (Table 2 and 3). In our evaluation, the *user coverage rate* refers to the fraction of users from the test set for whom our approach (or the baseline approach, respectively) could predict the remaining answers in a dialog session. As mentioned in Section 2 depending on some parameters and the answers provided by a user, there exist cases in which our approach fails to make predictions for the user. These cases get significantly smaller with an increasing number of pre-answered questions. For this reason, Ta-

ble 2 and Table 3 present a list of results measured with a varying number of pre-answered questions.

In order to ensure comparability, we have used the same hyperparameters (determined via grid-search and 10-fold CV) for both approaches. The results presented in Table 2 and Table 3 (evaluation results achieved on *DS-A*) indicate that our approach achieves high accuracy scores on average. In almost all cases, these scores are significantly better than the baseline results. Table 4 summarizes the corresponding error rates of the predictions as well as the (standard) deviation of the prediction error for both datasets. These values lie in a reasonable small value range and can also be considered as stable regardless of the number of pre-answered questions. This further supports our observations regarding the prediction accuracy.

Apart from these results, the most promising outcome of our evaluation is the high effectiveness of our approach in terms of the *user coverage*. It is obvious that the standard collaborative filtering baseline approach requires a lot of user input in terms of pre-answered questions (i.e., initial questions) in order to predict all remaining answers for most users. In sharp contrast to that, our approach is able to strongly increase the user coverage rate which allows the algorithm to produce correct predictions for most users quite early in a session. For example, accurate predictions can be made for half of the users after 35% of the questions (in case of DS-A; 50% in case of DS-B) have been answered by the users.

A comparison between Table 2 and Table 3 reveals that our approach heavily depends on the amount of training samples (historic dialog sessions from different users). As can be seen in Table 3 (*DS-B*), the first predictions could be made after around half of the questions were asked. The main reason for this phenomenon is directly related to the structure of dataset *DS-B*. As mentioned at the beginning of this section, *DS-B* consists of 78 questions and only 32 dialog sessions. Given such a large number of predicted answers and such a small number of training samples (80% of 32 dialog sessions results in 25 or 26 samples), the user coverage as well as the accuracy results can still be regarded as promising. In particular, this stands out when the achieved user coverage rates of our approach are compared with those of the baseline approach.

In general, the evaluation results of our approach reveal that our approach represents a proper means to efficiently decrease the number of pre-answered (initial) questions for most respondents / users without having to lose much accuracy. This means that in many practical situations, the developed prediction system allows these respondents to leave the dialog unfinished (having less questions answered). The more training data (i.e., historic session dialogs) are made available to the prediction system, the higher the chances are that very early predictions can be made for most users (see results achieved on DS-A which contains many session dialogs of different users). This is particularly essential for very complex configuration systems which require many user constraints. In such cases, long dialogs with many questions would be needed to gather the necessary user preferences. By applying our prediction approach, these dialogs could be enormously reduced once a sufficient amount of complete session dialogs has been collected.

Hence, the results also clearly show that our approach reaches its full potential once a sufficient amount of data is available. First empirical insights and observations (also with further datasets) indicate that a sufficiently high number of complete dialogs (at least more than 50 complete dialogs of different users) and a sufficiently high number of questions (at least more than 15 questions) is required in order to sharply reduce the number of questions by keeping the uncertainty as well as the error rates at a low level. Moreover, the used (i.e., learned via grid-search) hyperparameters (thresholds) have proven to be a good compromise between achieving a good prediction quality and reducing the length of a questionnaire for the majority of respondents.

Finally, another important observation that needs to be mentioned is related to the user coverage rates. By taking a look at the evaluation results achieved on both datasets, it can be observed that the user coverage rate initially increases with the number of pre-answered questions, but stops to grow fast after a certain level is reached (approximately after 65% of pre-answered questions have been asked in case of both datasets). This relates back to the phenomenon of rare cases described in Section 2 where our prediction approach fails to generate predictions due to our strict threshold parameters. An effective way to counteract this phenomenon is to collect more training data. Moreover, the threshold parameter could be increased (relaxed constraint) which would, however, deteriorate the average prediction accuracy. Another approach to tackle this phenomenon is explained in the remainder of this section.

Our empirical observations show that starting from this level of pre-answered questions (in case of both datasets, around 65% of pre-answered questions) model-based collaborative filtering algorithms (e.g., *matrix factorization* [9]) can also be successfully applied. Typically, such algorithms are known to work well on very large and sparse datasets [3]. Unlike our approach, further evaluations with *matrix factorization* (MF) indicate that MF is less applicable to tackle the cold-start scenario (i.e., make very early and accurate predictions after only a few pre-answered questions are available) since it would require large amounts of training data. Although these evaluations are still ongoing and, hence, part of future work, we want to provide some first insights. Our preliminary results show that the accuracy of MF was even worse when compared to our baseline approach which was the reason why we did not choose MF as a baseline in the evaluation presented in this paper. However, our still ongoing evaluation shows that in a *later* scenario (i.e., after more pre-answered questions are available) MF seems to attain comparable accuracy results also for candidates where our approach (presented in this paper) as well as the baseline approach fail to predict all answers of the remaining questions. This is due to the reason that even though we operate on quite small datasets, these datasets can be regarded as complete datasets (without missing data) where *only* the remaining part of the active dialog session is missing (approx. 35% of questions that have not yet been answered) and needs to be estimated by MF. In contrast to the approach presented in this paper, MF could thus be used as a fallback solution for those rare cases as it can deliver predictions for all users immediately without asking any further questions once they have already pre-answered more than half of the questions. Thereby, the prediction error of MF can be expected to reach a similarly low level when compared to our approach. Hence, predictions can be generated for all active dialog sessions and a maximum user coverage rate of 100% can be achieved which means that predictions can be made for all users once more than half of the questions have been answered by these users. Moreover, model-based algorithms based on collaborative filtering (such as matrix factorization) are also suitable for large datasets to address scalability issues in an efficient manner.

4 Related Work

To the best of our knowledge, there exists a relatively limited number of solutions to facilitate preference elicitation dialogs in the context of configuration systems. In particular, some research has been conducted which investigates the use of recommendation techniques to

Pre-answered questions	**User coverage**		**Avg. Accuracy (answer combinations)**		**Avg. Accuracy (individual answers)**	
	Approach	*Baseline*	*Approach*	*Baseline*	*Approach*	*Baseline*
4 (20%)	32%	5.22%	0.945	0.908	0.979	0.968
5 (25%)	36%	8.26%	0.914	0.894	0.968	0.961
6 (30%)	48%	11.02%	0.882	0.888	0.956	0.959
7 (35%)	64%	13.92%	0.888	0.884	0.960	0.956
8 (40%)	64%	17.00%	0.888	0.882	0.960	0.955
9 (45%)	68%	19.99%	0.884	0.875	0.958	0.952
10 (50%)	72%	23.23%	0.890	0.873	0.960	0.951
11 (55%)	72%	26.26%	0.890	0.869	0.960	0.950
12 (60%)	76%	29.30%	0.896	0.865	0.962	0.949
13 (65%)	84%	32.96%	0.885	0.862	0.961	0.948
14 (70%)	84%	36.36%	0.885	0.861	0.961	0.947
15 (75%)	84%	40.58%	0.885	0.857	0.961	0.946
16 (80%)	84%	45.45%	0.885	0.855	0.961	0.946
17 (85%)	84%	50.77%	0.885	0.856	0.961	0.947
18 (90%)	88%	58.74%	0.891	0.858	0.963	0.947
19 (95%)	88%	70.31%	0.891	0.860	0.963	0.947

Table 2. Accuracy results achieved on the first dataset (DS-A) after each iteration.

Pre-answered questions	**User coverage**		**Avg. Accuracy (answer combinations)**		**Avg. Accuracy (individual answers)**	
	Approach	*Baseline*	*Approach*	*Baseline*	*Approach*	*Baseline*
39 (50%)	57.14%	1.64%	0.981	0.840	0.995	0.950
43 (55%)	71.43%	2.87%	0.985	0.843	0.996	0.951
47 (60%)	71.43%	4.00%	0.985	0.846	0.996	0.952
51 (65%)	71.43%	5.41%	0.985	0.851	0.996	0.953
55 (70%)	85.71%	7.24%	0.981	0.851	0.993	0.953
58 (75%)	85.71%	8.77%	0.981	0.852	0.993	0.954
63 (80%)	85.71%	11.79%	0.981	0.855	0.993	0.955
66 (85%)	85.71%	14.33%	0.981	0.856	0.993	0.955
70 (90%)	85.71%	18.97%	0.981	0.860	0.993	0.956
74 (95%)	100.0%	28.50%	0.912	0.863	0.973	0.957

Table 3. Accuracy results achieved on the second dataset (DS-B) after each iteration.

Pre-answered questions		Mean prediction error		Standard deviation of prediction error	
DS-A	*DS-B*	*DS-A*	*DS-B*	*DS-A*	*DS-B*
4 (20%)	-	0.055	-	0.058	-
5 (25%)	-	0.086	-	0.103	-
6 (30%)	-	0.118	-	0.120	-
7 (35%)	-	0.112	-	0.106	-
8 (40%)	-	0.112	-	0.106	-
9 (45%)	-	0.116	-	0.104	-
10 (50%)	39 (50%)	0.110	0.019	0.105	0.011
11 (55%)	43 (55%)	0.110	0.006	0.105	0.011
12 (60%)	47 (60%)	0.104	0.006	0.105	0.013

Table 4. Error rates of predicted answers after each iteration.

proactively support users in configuration scenarios for the sake / purpose of preference elicitation [2, 4, 16]. Examples thereof include the recommendation of features as well as feature values [4, 16], the proposal of reasonable relaxations to repair inconsistencies of user requirements in situations where no solution can be identified [7], and the reuse of previous cases (that resemble the current problem) to determine new configurations [17]. Tiihonen and Felfernig [16] discuss recommendation techniques which help to complete configurations in ongoing configuration sessions. Moreover, Falkner et al. [6] propose recommendation techniques that interpret the user preference elicitation process as a dialog session where users are asked to specify feature values (i.e., the *user requirements / constraints*) for features presented in a pre-ranked order. The proposed approaches exploit different techniques to select and rank the features that are presented to the user and based on a user's selected values, explanations are shown to mark potential inconsistencies.

Further research is related to the automated support of active questionnaire dialogs. Stettinger et al. [15] introduce an e-learning environment which exploits different recommendation techniques following the purpose of intelligent question answering. The used techniques identify the most relevant learning contents and multiple-choice questions for a user that have to be repeated in future learning sessions. The presented approach is particularly useful to counteract forgetting processes in the context of online learning.

A major strength of the approach presented in this paper is the predetermination of a reduced set of textual questions with multiple responses (i.e., multiple answers can be selected for each question) whereas existing approaches focus on the assignment of single values to given variables. This stands in stark contrast to all existing approaches of related research discussed above. Another major difference that exists between our approach and most existing approaches, is related to the nature how the chronological order of the proposed initial questions is determined. Apart from that, available solutions focus on directly asking users to assign preferred values to variables / features instead of indirectly eliciting the preferences by asking more detailed verbose questions. Moreover, most existing approaches do not go beyond the level of ranking or feature selection and thus are not able to complete an ongoing session dialog. Due to the very simple nature of most approaches, these approaches are less suitable for drawing reasonable comparisons between these approaches and our approach. In particular, this applies above all to the measurement of the prediction accuracy (see Section 3), which needs to be assessed on the level of *multiple-choice* questions (instead of single-choice questions as presented in previous work).

5 Conclusion & Future Work

Conclusion. Configuration systems follow a constraint satisfaction problem (CSP) oriented approach to find or generate suitable solutions for a user. Due to the nature of CSP-based approaches, constraints have to be defined which should precisely reflect the domain knowledge as well as the individual preferences of the user. In particular, the elicitation of user preferences typically requires a lot of input of the users which can be collected by asking specific questions in a dialog-based style. In this paper, we introduced a new concept to facilitate the preference elicitation process. The major objective of our approach is to address user-related cold-start problems by reducing the length and duration of preference elicitation dialogs. This helps to significantly raise the efficiency of the overall configuration process. The evaluation results show that our approach has the potential to significantly reduce the number of pre-answered questions in most dialog sessions. According to our results, only between 35% and 50% of questions have to be explicitly answered by most users before our approach can predict the answers of all remaining questions. This makes our approach well suited to guide new users through dialog-based preference elicitation sessions for most complex configuration scenarios which involve an extensive variety of variables. One threat of validity is related to the limited explanatory power of our evaluation results. The evaluation of the presented approach was conducted with two smaller datasets. A reevaluation with different and larger datasets may lead to small deviations of the results presented in this paper. Moreover, it is important to point out that the applicability of our approach can also be considered to be somehow limited to more complex configuration scenarios and domains where users want to complete a configuration session as fast as possible.

Future Work. The currently applied methodology for the selection of the next relevant question is based on dissimilarity. In future evaluations it is planned to evaluate additional methodologies based on further statistical metrics with regard to the prediction quality of the remaining answers in preference elicitation dialogs. Moreover, the support of *system constraints* (*background knowledge*) represents another important issue for future work. The strict consideration of system constraints requires to rule out some answer-combinations in a preference elicitation dialog, since such answers would immediately introduce inconsistencies and conflicts. Within the scope of future work, we want to extend our existing approach to automatically extract rules from the system constraints and include these rules to disallow certain answers in a dialog-based session. Furthermore, further ideas for future work are related to apply automatic diagnoses and present explanations on how to resolve a conflict whenever users try to select a disallowed answer combination (e.g., an answer combination that would lead to the variable assignment *ElectricalDrive=YES* $\wedge$ *gearbox=MANUAL* would violate the existing knowledge base of a car configuration system since manual transmission is not feasible for electrically powered cars). Besides of the benefits our approach has for configuration-based systems, the presented approach can also be applied in questionnaire- or exam-related scenarios. Due to the fact that a significantly smaller amount of questions has to be answered by users in a multiple-choice dialog, the given answers are most likely of higher quality. This results from the fact that the respondents tend to lose concentration or provide sloppy feedback after having answered a large number of questions. Additionally, the reduction of the amount of asked questions can lead to a significant increase of the numbers of respondents in real-world surveys or questionnaires, since less time is needed and thus more users can be motivated to participate. In the scope of future work, we plan to conduct further investigations and studies to ground the mentioned hypothesis. Other application domains of the mentioned techniques could be a precise forecast of personal opinions related to societal / political / social / economic topics even though the person is not forced to answer questions which focus on these aspects.

REFERENCES

[1] C. C. Aggarwal, *Knowledge-Based Recommender Systems*, 167–197, Springer International Publishing, 2016.

[2] L. Ardissono, A. Felfernig, G. Friedrich, A. Goy, D. Jannach, G. Petrone, R. Schafer, and M. Zanker, 'A framework for the development of personalized, distributed web-based configuration systems', *AI Magazine*, **24**(3), 93, (Sep. 2003).

[3] D. Bokde, S. Girase, and D. Mukhopadhyay, 'Role of matrix

factorization model in collaborative filtering algorithm: A survey', *Journal of Advance Foundation and Research in Computer (IJAFRC)*, **1**, (2014).

[4] R. Cöster, A. Gustavsson, T. Olsson, and A. Rudström, 'Enhancing web-based configuration with recommendations and cluster-based help', in *In Proceedings of the AH'2002 Workshop on Recommendation and Personalization in eCommerce*, (2002).

[5] S. P. Erdeniz, A. Felfernig, R. Samer, and M. Atas, 'Matrix factorization based heuristics for constraint-based recommenders', in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, SAC '19, p. 16551662, New York, NY, USA, (2019). Association for Computing Machinery.

[6] A. Falkner, A. Felfernig, and A. Haag, 'Recommendation technologies for configurable products', *AI Magazine*, **32**(3), 99–108, (Oct. 2011).

[7] A. Felfernig, G. Friedrich, M. Schubert, M. Mandl, M. Mairitsch, and E. Teppan, 'Plausible repairs for inconsistent requirements', in *Proceedings of the 21st International Jont Conference on Artifical Intelligence*, IJCAI'09, p. 791796, San Francisco, CA, USA, (2009). Morgan Kaufmann Publishers Inc.

[8] A. Felfernig, J. Spöcklberger, R. Samer, M. Stettinger, M. Atas, J. Tiihonen, and M. Raatikainen, 'Configuring release plans', in *Proceedings of the 20th Configuration Workshop, Graz, Austria, September 27-28, 2018.*, pp. 9–14, (2018).

[9] Y. Hu, Y. Koren, and C. Volinsky, 'Collaborative filtering for implicit feedback datasets', in *2008 Eighth IEEE International Conference on Data Mining*, pp. 263–272, (2008).

[10] D. Jannach, L. Lerche, and M. Zanker, *Recommending Based on Implicit Feedback*, 510–569, Springer International Publishing, Cham, 2018.

[11] G. Jawaheer, M. Szomszor, and P. Kostkova, 'Comparison of implicit and explicit feedback from an online music recommendation service', in *Proceedings of the 1st International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, HetRec '10, p. 4751, New York, NY, USA, (2010). Association for Computing Machinery.

[12] D. Oard and J. Kim, 'Implicit feedback for recommender system', *Proceedings of the AAAI Workshop on Recommender Systems*, (07 2000).

[13] D. Sabin and R. Weigel, 'Product configuration frameworks-a survey', *IEEE Intelligent Systems*, **13**(4), 4249, (July 1998).

[14] Y. Salem, J. Hong, and W. Liu, 'History-guided conversational recommendation', in *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14 Companion, p. 9991004, New York, NY, USA, (2014). Association for Computing Machinery.

[15] M. Stettinger, T. Tran, I. Pribik, G. Leitner, A. Felfernig, R. Samer, M. Atas, and M. Wundara, 'Knowledgecheckr: Intelligent techniques for counteracting forgetting', in *ECAI 2020 - 24th European Conference on Artificial Intelligence, including 10th Conference on Prestigious Applications of Artificial Intelligence, PAIS 2020 - Proceedings*, Frontiers in Artificial Intelligence and Applications, pp. 3034–3039. IOS Press, (August 2020).

[16] J. Tiihonen and A. Felfernig. Towards recommending configurable offerings, 2008.

[17] H.-E. Tseng, C.-C. Chang, and S.-H. Chang, 'Applying case-based reasoning for product configuration in mass customization environments', *Expert Systems with Applications*, **29**(4), 913–925, (2005).