B. Sc. Thesis *

Questioning the Bot: Explanaitions for a Conversational Movie Recommender

Freie Universität Berlin

Institute of Computer Science

Human-Centered Computing (HCC) Research Group

Björn Bendict Heyder

Date Submission: October, 1, 2021

Supervisor:

Prof. Dr. Claudia Müller-Birn†, Freie Universität Berlin, Germany

Prof. Dr. Eva Zangerle ‡, University of Innsbruck, Austria

Examiner:

Prof. Dr. Claudia Müller-Birn, Freie Universität Berlin, Germany

Prof. Dr. Eva Zangerle, University of Innsbruck, Austria

---

*Template Version 2.0 — 2021-07-14

†Department of Mathematics and Computer Science, Human-Centered Computing Research Group

‡Department of Computer Science

Eidesstattliche Erklärung

Ich versichere hiermit an Eides Statt, dass diese Arbeit von niemand anderem als meiner Person verfasst worden ist. Alle verwendeten Hilfsmittel wie Berichte, Bücher, Internetseiten oder ähnliches sind im Literaturverzeichnis angegeben, Zitate aus fremden Arbeiten sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungskommission vorgelegt und auch nicht veröffentlicht.

Berlin, den October 1, 2021

Björn Benedict Heyder

Summary

Explainable AI (XAI) and conversational recommender systems (CRS) have both seen a rise in recent years. However, explanations for CRS (XCRS) are still rare and limited. The use of a design framework for XAI revealed that result, context and counterfactual explanation are of particular importance for XCRS. The importance of counterfactual explanations, especially for end-users with no AI knowledge, is already identified in the field of XAI. However, to the best of my knowledge, this will be the first CRS that integrates counterfactual explanations. In this thesis, a Conversational Movie Recommender will be equipped with a counterfactual explanation system, that gets a suggestion from the user and answers why that film was not recommended and how the user preferences must be changed to get the film as a recommendation. Additionally a context explanation, in which the believed user preferences get stated as explanation and a result explanations that closer explains facts of the recommended movie will be implemented.

Zusammenfassung

Erklärbare KI (XAI) und Conversational Recommender Systems (CRS) haben in den letzten Jahren einen Aufschwung erlebt. Erklärungen für CRS (XCRS) sind jedoch immer noch selten. Die Verwendung eines Designprozesses für XAI hat gezeigt, dass Ergebnis, Kontext und Kontrafaktische Erklärungen für XCRS von besonderer Bedeutung sind. Die Bedeutung kontrafaktischer Erklärungen, insbesondere für Endnutzer ohne KI-Kenntnisse, ist im Bereich der XAI bereits bekannt. Meines Wissens existiert jedoch kein CRS das kontrafaktische Erklärungen integriert. In dieser Arbeit wird ein Conversational Movie Recommender mit einem kontrafaktischen Erklärungssystem ausgestattet, das einen Vorschlag vom Nutzer erhält und beantwortet, warum der Film nicht empfohlen wurde und wie die Nutzerpräferenzen geändert werden müssen, um den Film als Empfehlung zu erhalten. Zusätzlich wird eine Kontexterklärung implementiert, in der die vermuteten Nutzerpräferenzen als Erklärung präsentiert werden und eine Ergebniserklärung, die die Fakten des empfohlenen Films näher erläutert.

Contents

List of Figures

List of Tables

1 Introduction

Conversational Recommender Systems (CRS) are helping the user to cope with information overload by finding items of interest. They support the user in a well-defined recommendation goal through a multi-turn dialogue. This dialogue tries to mimic a human sales conversation. With a preference elicitation phase and a recommendation and critiquing phase. This approach has been proven effective in approximating the user's preferences and providing an adequate recommendation. However while the system can build a good model of the user, the user cannot question the bot and develop an adequate mental model of the bot. This one-sided approach lacks a central part of human interaction; explanations. The need for explanations has been identified as a major challenge in Artificial Intelligence in recent years and the development of XAI methods has seen a huge spike. However, only a "few papers so far have studied the explanation issue specific to CRS" [7] and this besides the fact that CRS would profit particularly from implementing explanations. This work will therefore use the findings from recent XAI research to develop an explainable conversational recommender system (XCRS) in the field of movie recommendation.

1.1 Motivation

While the motivation for explanations varies depending on the actor (see Section 2.1.1), there is an intrinsic human need for explanations. Humans have a natural curiosity for the causal working of their environment and of the actors with which they interact. [1] They attribute beliefs and intentions to other humans as well as to machines and built a mental model of them. [12]. Understanding the system's decisions through a functioning mental model is central for trusting the system and knowing how to improve the system. The conversation of a classical CRS is solely aimed at helping the bot to create a good understanding of the user, by questioning the user about their preferences and asking for critique. The user on the other hand has no direct way of questioning the bot to create a mental understanding of the bot. The only way the user can create and update his mental model of the bot is through their input and the recommendation they receive. However, when an unexpected event occurs, like the recommendation of a movie that does not fit the user's preferences, the user needs to update his mental model. [1] Without explanations the user has no way of updating his model, leaving him with no functioning mental model of the system. Without a functioning model, the trust in the CRS is undermined, this will also lead to a lower acceptance of the bot and less willingness to further invest in the bot. A difference between expected and received recommendations also indicates a difference in believed preferences and actual preferences. Providing an explanation can serve as an information exchange point, that can increase the concordance between real and believed preferences and thereby help to find better recommendations. Explanations can also help to make the conversation more human-like and thereby more natural to the user. Humans constantly use explanations

when recommending movies [14]. So the integration of explanations into CRS can make the usage feel more natural for the user.

1.2 Research Goal

Besides all the advantages that explanations bring, they are not wildly used in current CRS. This might be due to the misconception that explanations are only necessary for high stake areas. However, the need for explanations is universal and is even more necessary in systems, like CRS, that rely on trust, and a shared belief. The current XCRS, as closer discussed in section 3.4 are currently lacking a structured human-centred approach for identifying the user needs and providing the appropriate explanations. The goal of this thesis is to use the design process of Liu et al. [10] [11] to create a CRS that is equipped with human-centred explanations. For this, a CRS for movie recommendation will be created, which will be used as a basis. In the first phase of the design process, possible questions a user might have when using a CRS for movie recommendations will be heuristically gathered and analysed. This will answer what questions a CRS in the domain of low-stake recommendation for lay users, like movie recommendation, should answer. This analysis revealed that result explanations, context explanations and counterfactual explanations are the most important type of explanations that should be answered. Result explanations are not providing the user with insights about the bot, but rather about the recommended item. Such a type of explanation is used in almost every CRS and often not even considered as an explanation. Context explanation provides the user with an overview of the beliefs the system has about the user's preferences, such form of explanation is used by a few bots [6] [13]. Counterfactual explanations answer the question why not P. Such questions answer the causal relationship of the recommendation process and provide the user with insights into the working of the CRS. While there are examples of CRS that provide causal explanations, they are limited to answering only why questions [13]. However, as Miller suggests in his work about human-centred explanation [12], human causal explanations are inherently counterfactual. This work will therefore use counterfactual explanations to provide the user with causal explanations. This will, to the best of my knowledge, be the first CRS that provides a counterfactual explanation. This work and the developed XCRS shall furthermore serve as a basis for further research into explanations in CRS.

1.3 Structure

The work is structured into 8 chapters, including this introduction (Section 1). In the next chapter (Section 2), an overview of the two subjects concerning this work; explainable AI (XAI) and conversational recommender systems will be given. Chapter 3 will present examples of current explainable conversational recommender systems (XCRS) alongside an analysis of the current research gap. The 4 chapter will introduce the design

process of Lia et al. as a solution for a more structured and substantive way of creating XAI. Based on this framework chapter 5 will provide a heuristic gathering of user explanation needs and analyse them together with their possible answers. In chapter 6 the implementation of the bot, the recommender and the explanations will be presented. Chapter 7 will discuss the findings of this fork and chapter 8 will conclude the thesis with a remark on the limitations and a forecast about possible future work.

2 Theoretical Foundations

This part presents the theoretical background of the subjects concerning this thesis. The first part will give an overview of the field of explainable Artificial Intelligence (XAI). The second part will introduce conversational recommender systems.

2.1 Explainable AI

The field of Artificial Intelligence has seen a steady rise with better predictive performance due to the exponential increase of data and computational power. The field of AI is predicted to become even more important in the coming years, with a significant increase of both the investment in AI and revenue generated by AI. [1] With the use of AI in every aspect of life, and especially in high-stake areas like healthcare or finance, the need to verify, interpret and understand the reasons behind the system's decision become evident (a closer explanation for this need can be found in the next Section 2.1.1). This field of explainable AI (XAI) tries to make AI systems interpretable and explainable. The terms "interpretable" and "explainable" do not have an agreed-upon definition. However broadly speaking one can use them interchangeably, as it will also be done in this thesis. Broadly defined a system is interpretable/explainable if its operations can be understood by a human. That means in the context of a decision-making system like AI that the decision and the reasons behind it, have to be understandable by a human. In the following a closer analysis of why explanations are so important will be given (Section 2.1.1), then an overview of the taxonomies of the methods and explanations used in XAI (Section 2.1.2). This overview of XAI is based on the literature review of Carvalho et al. [1].

2.1.1 Motivations for Explanations

The term XAI was first coined by Lent et al. [9]. Since then there was a sporadic interest in XAI in the 1970s, 1980s and 2000s each following the waves of expert systems, neural networks and recommender systems. In the 2010s the focus was put on predictive power. However, in the last decade explainability has seen a new wave of interest, following the penetration of complex and nontransparent ML systems in many domains of interest. This new awareness is shown in the public sphere where several new guidelines and reports by the US, EU and many others have stressed the need for explainable AI to increase the transparency, trust and accountability of AI systems. The industry, including big players like Google, Facebook and IBM, are reacting to this trend and by focusing on AI products that can explain their decisions. In the scientific field, the interest in XAI is not limited to Data Science. It also includes the field of Human-Computer Interaction, which works on empowering the user in the decision process, and the field of Human Science, which studies how humans produce and understand explanations. All of these actors, together with the end-users are fuelling the need for interpretability in AI.

For the public and the economy, fairness and safety are the main reasons for integrating interpretability into AI. This need is especially strong in high-stake areas, where wrong predictions have severe consequences. Examples for such areas would be the criminal justice system, where a program has incorrectly denied parole for ethnic reasons [18], or the healthcare and finance sector. In such systems, there is a high risk that systems incorporate biases, either based on ethnicity, gender, age or other protected properties. In a black-box system, with no explainability, the wronged person would have no ground for appeal. To counter this risk, regulations have been enacted on machine-based decisions. The European General Data Protection Regulation gives every data subject the right to an explanation of algorithmic decisions [1], and companies almost everywhere are accountable if they discriminate against a user based on protected properties. Also in decisions that do not impact an individual directly, there are safety concerns that can be addressed through explainability. XAI can help test, audit and debug a system by analysing the explanations given by the system through human reasoning. As an example, an image classification program for autonomous driving might state that it detects bikes based on two wheels. In such cases, the engineers might want to think about edge cases like unicycles or cases where one wheel is concealed.
For the scientific community, the main concern lays in the acquiring of new knowledge and the discovery of new scientific findings. With the use of black-box machine learning in the analysis of big data new scientific findings are possible. However, the knowledge of these findings is quite limited, because the model becomes the source of knowledge, but the deeper meaning of the findings stays hidden to the researcher. XAI can extract the causal relationships encoded in the system and provide them to the researcher.
For humans in general explanations are deeply embedded in the human way of thinking and every end-user has a natural curiosity for causal correlations that can be satisfied by explanations. This curiosity is not limited to high-stake applications, it concerns every aspect in which a system impacts a human. People attribute beliefs and intentions to computer systems and develop a mental model of the system. Therefore, they need an explanation if the expectation of the system's behaviour is inconsistent with reality. The use of explanations is used in human interaction to create shared meaning and can thereby not only increase trust but also social acceptance. Humans are driven by intuition and reasoning and explanations provide AI systems with a way of communicating with humans on this level. Thereby explanations provide humans with the opportunity to stay in the loop with the AI system, to reason with its decisions and at the end to use it as a tool, rather than submit to its decisions.

2.1.2 Taxonomies of XAI methods

In the following, an overview of the taxonomies of both the methods and explanations concerning XAI will be given. This overview will help to define the scope of XAI and help to classify and analyse different XAI methods and explanations. An overview of the taxonomy of XAI methods can be seen in Figure 1.

[1] General Data Protection Regulation Article 71

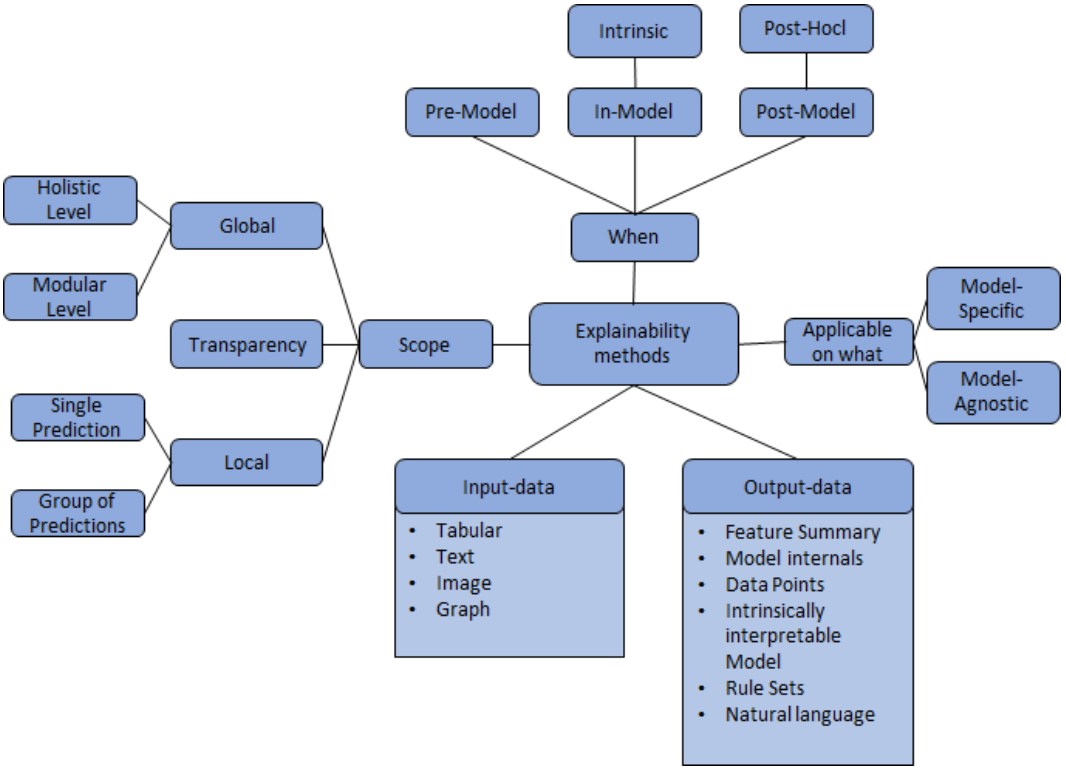

Figure 1: Overview of the Taxonomy of XAI methods

When is the XAI method applicable in regards to the building of the ML model? - before *Pre-Model*, during *In-Model* or after *Post-Model* is the first dimension on which we can divide XAI methods.
Pre-model methods are used before the ML model is trained, which means they analyse and explore the data that is used to later train the model. Thereby pre-model explanations are closely related to data interpretability, data analysis and data visualisation. The goal of such systems is to better understand the data used and to find biases or inconsistencies before the model is trained.
In-Model interpretability concern ML models that are *intrinsically* explainable. This means that only ML models which are interpretable by themselves either through sparsity, monotonicity, causality or physical constraints are used. Such systems are also called transparent and answer the question of how the model works.
Post-model methods are applied after the ML-model is trained, they do not impose any constraints on the model and are used to *post-hoc* analyze the trained model. Such methods answer the question of what else the model can tell us.

Model-specific vs. model-agnostic is concerned with the question of what kind of ML model the method can be used on.
Model-specific methods can only be used on specific sets of ML models. For example, the

interpretation of the weights in a linear model is model-specific. Most of these models are used on intrinsically explainable models.
Model-agnostic methods can be used on every ML model. They analyse pairs of input and output to generate relationships between them and thereby anticipate the working of the model without accessing the inner structure. Such models have to be used Post-Model, however, they can still be used on intrinsic explainable models.

The *scope* of the explanation, which means the portion of the prediction process the method claims to explain, is another dimension that classifies XAI methods.
Algorithm transparency explains how the algorithm creates its model. This means it only explains the general working of the algorithm and not the specific trained model or any prediction. Such explanations are easy to give because only knowledge about the algorithm itself is necessary, however, the value of such explanations is limited.
Global model interpretability aims to give explanations on how predictions are made by the model.
On a *Holistic Level* the entire model needs to be comprehensible at once. This means that the trained model, the fundamental algorithm and the data have to be completely integrated into the explanation. Such an explanation would answer on a complete level how the trained model makes predictions. However such explanations are in practice almost impossible to achieve because humans can not grasp the full parameter and feature space of most ML models. So in practice, the completeness of an explanation has to be traded for human-understandable simplicity.
Modular level explanations are reached when this trade is done. Modular level explanations only try to answer how a specific part of the model influences the prediction. Such explanations can only be given for at least partly intrinsic models. It is also important to note that such explanations usually do not account for feature interaction.
Local model interpretability zooms in on a *single prediction* or *group of predictions* and only tries to explain these. For this, the system zooms into the single instance that produces the prediction and approximates a simple model that sufficiently explains it.

The *Input Data* used for the ML is another way to differentiate between XAI methods. While many methods, and especially agnostic methods, work on multiple Input Data. Most work better with some data types than with others. The most important forms of Input Data are structured tabular data, text, images and graphs.

The *Result* that can be produced by the methods is another way of classificiation.
In a *Feature Summary* a statistic value, like importance, is provided for every feature. Such summaries are either done in tabular or visual form depending on the type and scope of values.
Model Internals can be used in intrinsic models as explanations. In some cases, like with the weights of a linear model, they are simultaneously Feature Summaries.
Surrogate Intrinsically Interpretable Model is a method in which an intrinsically interpretable model is approximated from the black-box model and then returned as an explanation. *Data Points* are often used with image Input Data. They are example-

based and return Data Points to explain their decision.

The last dimension on which the different methods can be divided is the kind of explanation. In the following, a look at the properties they possess will be taken. For this two sets of properties of explanations are presented. In the next section on design frameworks, a lock at another approach of dividing explanations and explanation methods along the line of the questions they answer will be taken.
Robnik-Sikonja et al. [17] present 9 properties that an individual explanation has:

- *Fidelity*: How well does the explanation approximate the prediction? Probably the most important property, because an explanation with low fidelity is meaningless.
- *Accuracy*: How accurate is the explanation for unseen data? When the ML model has low accuracy, low accuracy of the explanation might be fine. In a model with high accuracy, this metric is closely related to fidelity.
- *Consistency*: How similar are the explanations between two different models that have been trained on the same task and output similar predictions? Normally high consistency is desirable, however when the two models use different features and output the same data this difference should be reflected in the explanations and thereby in a low consistency.
- *Stability*: How similar are the explanations for the similar instances with similar predictions?
- *Comprehensibility*: How well do humans understand the explanation? It is one of the most important properties because it takes the receiver of the explanation into account. However, it is also subjective and highly dependent on the context and the personality of the receiver.
- *Certainty*: How confident is the model about a prediction?
- *Importance*: How important is the feature used in the explanation?
- *Novelty*: How far away is the instance from the distribution of training data? A high novelty might indicate a low certainty.
- *Repressiveness*: How many instances are covered by the explanation? Is the method used globally or locally?

2.2 Conversational Recommender Systems (CRS)

With the increase of information and options in almost every aspect of life, recommender systems that can filter through this information and present the user with a set of interesting items become more important. Such systems often take the approach of a one-shot interaction process, in which the system gives the user a one-time recommendation based on the user profile. While these systems are immensely popular, they come with a

set of limitations. Especially in situations where the system has no prior knowledge about the user (cold-start problem) or where the prior interaction of the user does not reflect his current preferences, like in high-involvement products that are purchased only every couple of years. Another problem of these one-shot approaches are context-dependent areas in which the recommendation is heavily influenced by the daily context of the user (for example movie recommendation might be influenced by the current mood of the user). One different approach that tries to tackle these limitations are Conversational Recommender Systems (CRS). In such systems, the user and recommender are engaged in a conversation-like loop of preference elicitation, recommendation, and critiquing of recommendation/updating of preference until a suitable recommendation is found or the process is terminated by the user. Such systems take a human-centred approach to recommendations, trying to mimic the conversation with a salesperson. While first envisioned in the 1970s by Rich [16], their popularity came in recent years through the major advances in natural-language processing and the increased adaption of chat-bots and voice-based home assistants. In the following an overview of CRS based on the literature review of Jannach et al. [7] will be presented. Firstly the interaction modalities and secondly the general architecture, computational task and data of CRS will be reviewed.

2.2.1 Interaction Modalities

The interaction of the user with a CRS system can be analysed from three different perspectives: In-/Output; application environment and interaction initiative

The In- and Output Modalities can be divided into three approaches:

- Based on structured layouts, like forms and buttons.
- Based on natural language, in written and/or spoken form.
- Application-Specific Inputs and Outputs.

Both forms can be found exclusively or combined in a hybrid approach. The structured layout approach is extensively studied and has the big advantage, that it does not need NLP and the dialogue path can be pre-defined. The NL approach offers the most flexibility and can sometimes also support chit-chat, but it still struggles with the understanding of the users' utterances and the identification of their intents. Furthermore, the presentation of the recommendation can be quite difficult. One example in which a mouse-based button interface was compared with a spoken-NL interface for the critique input is ReConnect [3]. ReConnect is a Unit Critique Recommender for compact cameras. In their study from 2013 Grash et al. suggest, based on a user study of 80 students, that NL-interfaces can provide better recommendations with much fewer feedback cycles and argue that this is based on a higher mental effort for formulating NL commands. However, several studies suggest that a mixed approach, in which the NL interface is

augmented with buttons, leads to the best user experience [5] [13].

Finally, one should note that besides the classical approach of NL and form-based approaches new modalities for specific use-cases are being developed. This includes on the input-side non-verbal communication like posture, gesture, speech prosody and facial expression, as well as visuals inputs like pen gestures and inputs on maps. On the output side, there are several approaches of interactive maps and the use of embodied conversational agents with non-verbal behaviour. Furthermore, CRS in virtual 3D spaces are being experimented with.

The application environment and supported device can have a major impact on the design choice of the CRS. Especially the use of smart speakers as device often limits the in- and output modalities to a solely voice-based interaction. Also for classical web and smartphone approaches factors like screen size play a significant role in the design choice. The application device space of CRS is growing with the technological development of new devices. Besides the already mentioned smart speaker, there are also approaches of CRS for interactive walls in supermarkets, service robots in restaurants and voice-based driving-related recommendations, that all come with their own design opportunities and challenges.

The interaction initiative, meaning who is driving the conversation, can be user-driven, system-driven or mixed. In a system-driven approach, the user does not take a proactive role. They first get presented with questions to determine their preferences and then with the first recommendation. Which the user can then critique on a pre-defined or dynamically produced set of attributes. In a user-driven approach, the user asks and the system responds. While Jannach et al. "[...]question if we would call such an exchange a conversational recommendation[...]" and claim to not have found any paper regarding such a system besides a group recommender. But there exists another study by MusicBot [8] in which a mixed and a user-driven approach are compared. This study even suggests that user-driven approaches need less effort and produce a recommendation with the same rating as their hybrid approach. However, it should be noted that the hybrid approach created more diverse recommendations. Still, the main approach of almost all CRS is a mixed initiative. NL-approaches, with their higher flexibility for user influence, are tending towards the user-driven approaches, while chatbots, relying on pre-defined forms and buttons, tend towards the system-driven approach.

2.2.2 Architecture, Computational Tasks and Data

CRS are built highly modular, with every module being responsible for a different task. Depending on the functionalities some CRS components might be more complex than others, but the architecture is always the same.

The *input-processing* is especially important when an NL interface is used and when

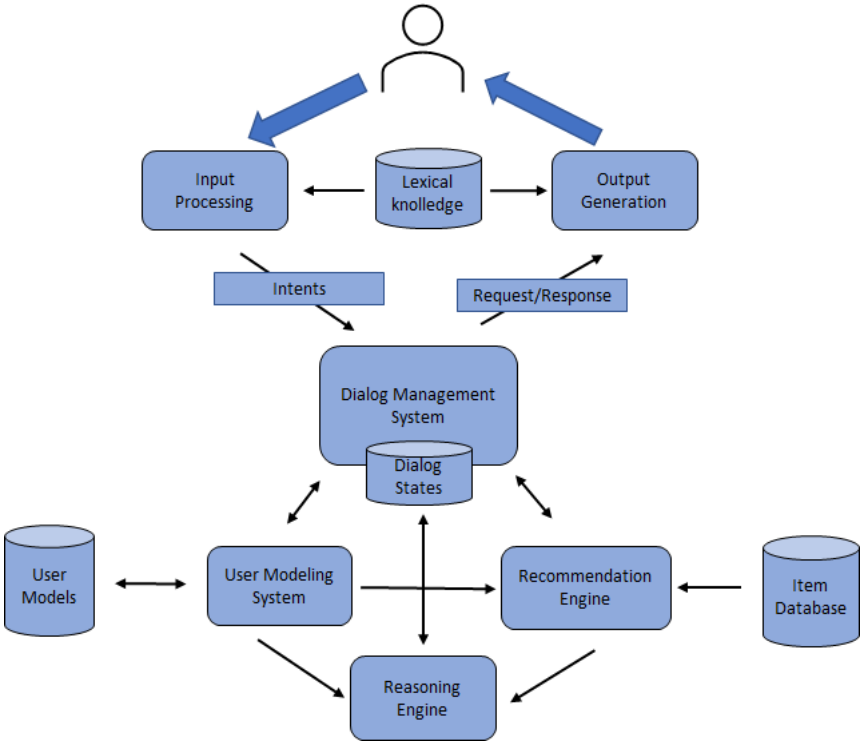

Figure 2: Basic architecture of a CRS after Jannach at al. [7]

the system is user-driven. The input-processing unit has to first extract the user intents and the values. A list of possible intentions can be seen in Table 1. Depending on the input modality the system might need to first use a voice to text conversation or in cases of a solely form-based interface the intend is pre-defined by the current form shown by the system.

The *Dialogue management system* gets the intends and corresponding values from the input processing unit and has to decide how to react to it. Depending on the interaction initiative the management system has to decide how to react to a user request (user-driven) or has to process the answer from the user and decide what to ask next (system-driven). Especially in system-driven CRS the management of the dialogue and the determination what to ask next is a big task. Most CRS conversations, and IAI-movieBoot's too, are divided into two parts. In the first the system acquires the user's preferences through questions, in the second the system recommends an item, and gives the user the possibility to critique, to ask for details, to ask for explanations, to revise the preferences or to accept/reject the recommendation. The goal for the system is to find a suitable recommendation in as few steps as possible. For this, most systems are based on a finite state machine. They keep track of the current dialogue state and from there have different transitions to other states. This can either be predefined by the developer or be learned from the data using machine learning. One major determinator

Intent name	Intent description
Initiate Conversation	Start a dialog with the system
Chit-Chat	Not recommendation related conversation
Provide Preferences	Share Preferences with the system
Revise Preferences	Revise previously made Preferences
Ask for recommendation	Obtain system suggestion
Obtain Explanation	Get reasons behind the recommendation
Obtain Details	Ask about details of recommended object
Feedback	Give feedback on the recommendation
Restart	Restart the dialog
Accept	Accept the given recommendation
Quit	Terminate the conversation

Table 1: Domain-independent intense after Jannach at al. [7]

in the transition of states and especially for the transition between the two conversation blocks is the current user preference.

The user preferences get managed by the *user modelling system*. For that most CRS use a slot filling approach. The developer predefines a set of attributes that can be filled with the values corresponding to the user's interest. The input processing unite provides these values and attributes together with the intent of stating/changing the preferences or criticising of a recommendation. The management system may proactively ask the user about missing slots in system-driven systems. When the system has acquired enough knowledge about the user it can start to recommend items. Nevertheless, some approaches just start with the most popular or most controversial item and only acquire the user's preferences through the critique.

The *recommendation engine* is responsible for the main task of a CRS, recommending a suitable item to the user. The main technical approaches to solve this task are collaborative, content-based, knowledge-based and hybrid approaches. Most CRS only use short-term preference information.
Additionally, a reasoning engine can give the user explanations about the recommendation in the next Section 3 a closer presentation of such methods will be given.

In the final step, the output of the system has to be generated. This part again is very different depending on the interaction modalities. The effort is especially high if NL is used as output. Then the output generation must translate the action into natural language. This can be done by predefined sentences or by machine learning. In both cases, the system needs lexical knowledge that should be broad, so that the system does not use the same phrases over and over again. In the case of the IAI-movieBot, several different base-sentences were predefined for each action which the system chooses randomly to sound natural.

3 Related Work

Besides the fact that explanations play a huge role for humans (Section 2.1.1) and CRS try to be as human-centred as possible, the literature review of Jannach et al. [7] found few papers that studied explanations for CRS. However, there are still some papers that studied different approaches to explanations in CRS, a selection of them will be presented in the following.

3.1 Context explanations

Jain et al. [6] identified a gap between users expectations and experience of chatbots. They found that users build a mental image of the chatbot that does not necessarily

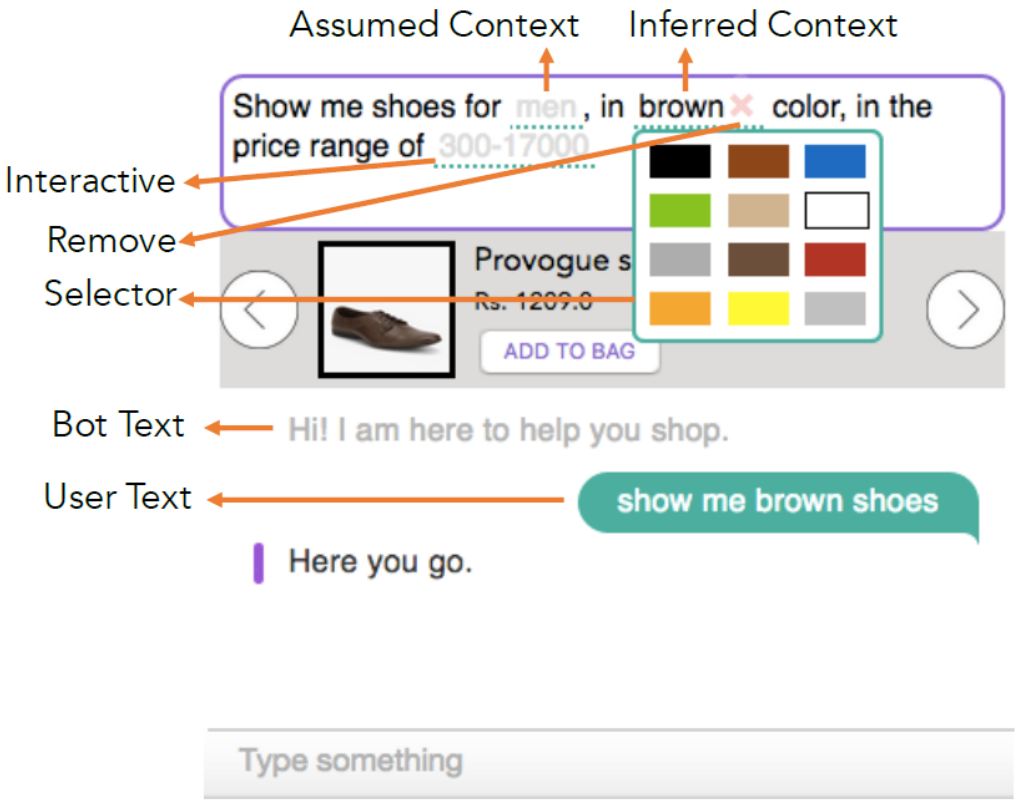

Figure 3: Design Interface of Convey [6]

match the beliefs of the system. Especially when complex tasks, like a recommendation, are wanted, as well as when the conversation gets longer, the beliefs of the system and the user can drive apart. Another problem are assumptions made by the system, for

example, based on the user's history or acquired through another statement. The author present Convey as a solution. Convey is an add-on for chatbots that adds an interface in which the system states its current state of beliefs. In a simple sentence, the system presents its current beliefs about what the user wants. It differentiates between assumed and inferred context. Additionally, the user can interact with the context in the sentence by clicking on it to either change or delete the belief (see Figure 3).

The authors conducted a user study in which two CRS for shoes were compared. The first one used no explanations, the second used Convey to explain its current beliefs. The results of the study should be considered with caution because it only had 16 participants, all with academic and often with a technical background. Besides these limitations, the study still clearly suggests that the additional explanations improved the system. While the log data is not that distinguishable; the participants using Convey needed a little bit more time and typed fewer messages (but had also additional interaction with the Convey-interface) the results of the experience ratings are much clearer. The participants rated the Convey version with explanations better on every metric (mental demand, physical demand, task success, effort, frustrating, use in future) than the default Bot without explanations (see Figure 4).

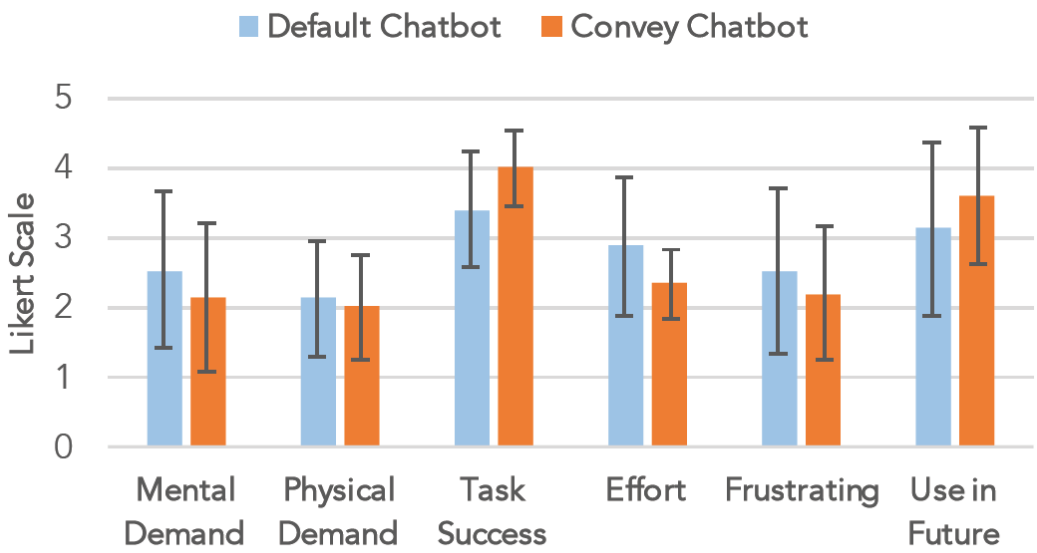

Figure 4: Perceived performance rated by the participants with Likert-scale (with standard deviation shown as error bars)[6]

3.2 Causal explanations

Narsucci et al. have done a study about possible ways to improve the user experience in CRS [13]. For this, they implemented a CRS for movie recommendations, that included

three forms of explanations. One is the possibility to explore user preferences. This is similar to Convey, but with the difference that the user has to actively ask for this explanation. The second form is a why-button which the user can press. This button answers the question of why the current recommendation was made. For this, the system uses the intrinsic explainability of the PageRank recommendation algorithm, which creates a graph that connects users liking of films with their properties. When asking why a recommendation was made, the algorithm can not only state the attribute that influenced the decision but also where this attribute came from. As an example, the system might say: "I recommended you Titanic because Leonardo DiCaprio plays in it, like in Inception.". The third way with which the system can make explanations is through giving details about the film. The authors conducted an extensive user study in which over 500 users tested 4 different UX designs. Unfortunately, the user study did not focus on the explanation, but rather on interaction modalities. For this, they implemented a clarification interaction in which the system asks for clarification when an unknown attribute is identified, by suggesting a similar written attribute. They also equipped the system with buttons that can both provide the user. The interface can be seen in Figure 5. The three findings they found are:

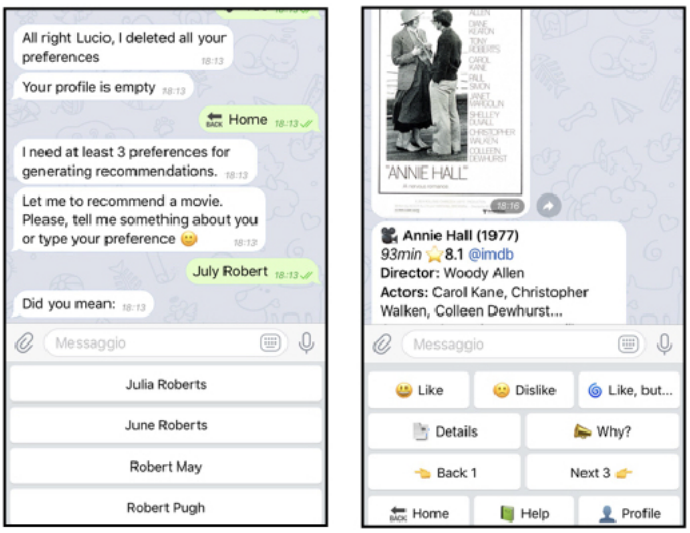

Figure 5: Caption

- In the cold-start problem it is better to start with the most popular item rather than with the most diverse.
- Preferences should rather be expressed on the movie, than on the movie property.
- Using written NL to express the preferences for an entity or property is better than using predefined buttons.

All these findings were both true for accuracy as well as interaction costs. The only thing that the user study revealed about the explanations is that half the users explored their user preferences at one point and for 13% of recommendations an explanation was asked for. The authors conclude therefore that explanations are features liked by the user.
Another example of "why" explanations can be found in Towards Explainable Conversational Recommendation by Chen et al. [2]. They also have identified the need for explanations in CRS and bidirectional user-model communication. The authors developed a CRS in which explanations play a central role. Besides the recommendation, the system provides the user with an explanation that states the two most important features of the recommendation. It, therefore, answers the why question proactively. While the work was mostly focused on the technical realisation of a deep learning approach to recommendation and explanation, they still provide a good example of how the answering of the why question through the most distinct feature can not only provide good explanations but also increase the accuracy of the predictions.

3.3 Social explanations

Pecune et al. developed a model of social explanations for a conversational movie recommender system [14]. For this, they analysed a corpus of human-to-human conversations concerning the recommendation and talk about movies. The explanations used in the conversations were annotated and clustered into 4 categories:

- Movie Feature (37%): These are general facts about the movie, like cast or plot.
- Third-Party Opinion (7%): These explanations are human-based and describe what other humans thought about the film. They can be both broad, about the people's general opinion, or specific, the opinion of one person.
- Personal Opinion (39%): This describes what the recommender thought personally about the film. The author divides this category again into positive, analytical and structured.
- Personal Experience (17%): This part can either be very specific like the comparison with other movies or anecdotal and logistic in which the personal experiences watching the film are explained (for example I watched it in the cinema and came too late).

Based on these findings they developed a human-like CRS that uses voice to communicate and is humanised by an animated virtual character. Like Narsucci et al. they also use the PageRank algorithm for the recommendations. They tested three explanation approaches. Either preference-based explanations generated by the recommendation engine, social-based by providing the user with randomly chosen explanation, based on the observed distribution, or no explanation. They additionally tested all three approaches

with random recommendations and personalized recommendations (based on the recommendation algorithm). The authors conducted a user study through MecanicalTurk in which 60 interactions (10 for each combination) were recorded. The use of the personalized recommendations had, as expected, a positive impact on both the quality of the conversational agent and the quality of the interaction. The use of an explanation was both for the quality of the conversational agent as well as of the interaction in general positive. Social explanations helped the user in learning more about the recommendation. It also helped the user to feel more attracted and in sync with the system. The users additionally had the feeling that the system is more interested in them. However, the personalised explanations were better in helping the user to understand the system.

3.4 Research gap

Taking a closer look at the here presented examples of explanations in CRS three points become evident. First, there is a clearer need for explanations in CRS. All three studies showed that explanations can help to achieve the recommendation task and make the process easier for the user and increase the overall satisfaction. However as Janach et al. already stated in his survey of CRS; there are "few papers so far [that] have studied the explanation issue specific to CRS"[7], which is also reflected in the limitation of related works. So there is a clear gap between the usefulness of explanations in CRS and the lack of research work concerning them. Furthermore, the work concerning explanations in CRS is often done without putting the user needs first. While the work of Pecune et al. [14] is a good example of first analysing human explanation patterns and orienting the explanations on them, it still lacks a clear design process. None of the CRS takes advantage of the recently developed frameworks in XAI. So there seems to be an even bigger need for explanations in CRS that are developed through a well-developed framework that takes the user needs in the centre. In the next Section 4 such a framework for the development of human-centred XAI will be presented and later used to close the here presented research gaps.

4 Design Process

Lia et al. [10] [11] present a framework for designing user-centred explanations for AI systems and bridging the gap to algorithmic centred XAI. They identified that the current approaches of categorizing explanations (Section 2.1.2) are mostly Data Science centred and do not take the perspective of the designer or end-user into account. In two papers, the authors take the approach of investigating the work by XAI industry practitioners to identify gaps between the algorithmic work of XAI and real-world user needs. For this, they focused on the UX designer perspective and conducted expert interviews with UX designers of different XAI projects from IBM. In their paper [10], they provide insights into the user needs for explanations in AI and argue that these needs should be the centre of the XAI design process. Based on the UX designer interviews, they extracted a list of questions a user can have about an AI system and grouped them into a question bank. The authors additionally found that only providing explanations

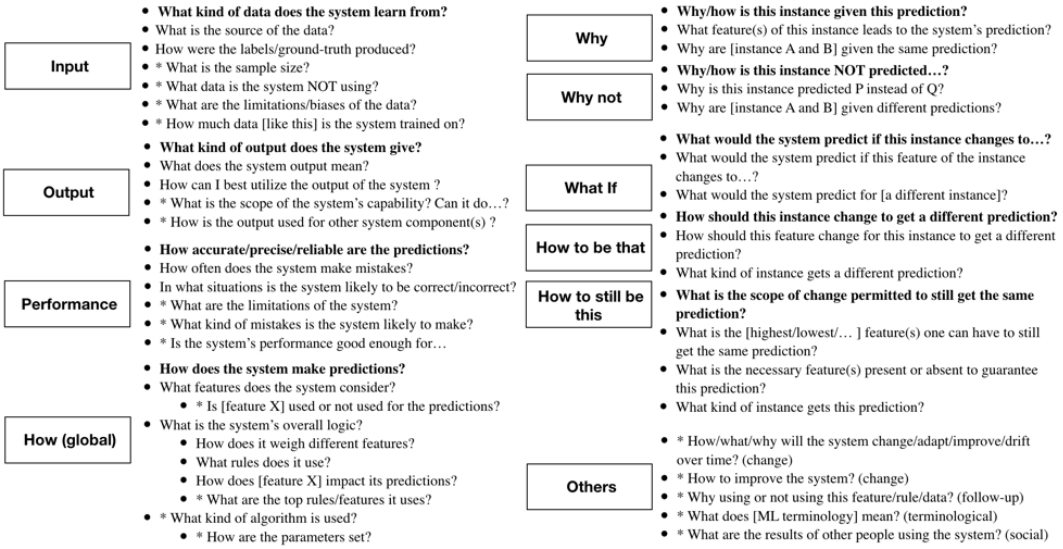

Figure 6: Question Bank identified by Lia et al. [10]

is insufficient if the user's motivations for the explanations are ignored. They therefore also identified several utility goals driving users demand for explanations:

- *Gain further insight or evidence* so that the user can learn from the system and develop a hypothesis about the causality. This can help to enhance the decision and even help the user to mitigate their own decision bias.
- *Appropriately evaluate the capability* by assessing the system's limitation and determine the overall system adoption.
- *Adapt usage or interaction behaviour to better utilize the AI.* Explanations help the user to understand what the system needs and can convince the user to invest in

the system

- *Improve the AI performance* by using explanations in a feedback loop, in which the user can correct the system when it makes a mistake.
- *Ethical responsibility* to provide reliable explanations to the user and affected individual, especially in high stake areas.

Besides the motivation, four other categories influence which type of explanation should be used:

- Usage point of the AI. At different points (for example during onboarding or encountering an abnormal result), different explanations are needed.
- Algorithm or Data Type: Especially different data (for example tabular data vs. images), but also different algorithms, invoke different questions.
- Decision context: Different types of context could be: time-sensitive, critical or complex.
- User Type: Characteristics, knowledge (AI and domain knowledge) and role.

The authors also found that explanations should have human-centred properties that are strongly aligned with Millers properties for human-friendly explanations (Section 2.1.2). At the end of the paper mentioned above, they provide a small overview of what the motivation behind each question was and from whom they are asked and in which situation.

- *Input*: Appropriately evaluate the capabilities. This is mostly needed in the onboarding phase by decision-makers and quality control.
- *Performance*: Appropriately evaluate the capabilities and limitations. Ranked less important in uncritical areas for users with no AI background. Criticised for not providing enough actionable information.
- *How-global model*: Gain further insights into the model, to better interact with it, to improve it and to evaluate the system's limitations. Requested by users in quality control with AI knowledge.
- *local prediction Why and Why not*: Often ranked as the most important question. Gives the user a very good insight into how the system works, by providing causal relationships, natural to human thinking.
- *inspecting counterfactual What if and How to be*: Not ranked high, but possibly because this explanation type is not widely adopted. Helps the user to gain further insight by testing different scenarios and to understand the limitations of the system.
- *Additional explanation needs*: Some of the explainable needs are not covered by the previous questions. These include understanding the changes and adaptation of AI, questioning why a feature is used, terminology questions and social explanations.

Lia et al. additionally identified the communication between designers and developers as an friction point and developed in their second paper [11] a mapping guide between user questions and XAI methods (Figure 7). This mapping guide serves as the transition

Question	Explanations	Example XAI techniques
Global how	• Describe what algorithm is used and what features are considered, if a user is only interested in a high-level view • Describe the general model logic as feature impact*, rules✦ or decision-trees• (sometimes need to explain with a surrogate simple model)	ProfWeight*✦•, Feature Importance*, PDP*, BRCG✦, GLRM✦, Rule List✦, DT Surrogate•
Why	• Describe what key features of the particular instance determine the model's prediction of it* • Describe rules✦ that the instance fits to guarantee the prediction • Show similar examples• with the same predicted outcome to justify the model's prediction	LIME*, SHAP*, LOCO*, Anchors✦, ProtoDash•
Why not	• Describe what changes are required for the instance to get the alternative prediction and/or what features of the instance guarantee the current prediction* • Show prototypical examples✦ that had the alternative outcome	CEM*, Prototype counterfactual✦, ProtoDash✦ (on alternative class)
How to be that	• Highlight features that if changed (increased, decreased, absent, or present) could alter the prediction* • Show examples with minimum differences but had a different outcome than the prediction✦	CEM*, Counterfactuals*, DiCE✦
What if	• Show how the prediction changes corresponding to the inquired change	PDP, ALE, What-if Tool
How to still be this	• Describe feature ranges* or rules✦ that could guarantee the same prediction • Show examples that are different from the particular instance but still had the same outcome	CEM*, Anchors✦
Performance	• Provide performance metrics of the model • Show confidence information for each prediction • Describe potential strengths and limitations of the model	Precision, Recall, Accuracy, F1, AUC Confidence FactSheets, Model Cards
Data	• Document comprehensive information about the training data, including the source, provenance, type, size, coverage of population, potential biases, etc.	FactSheets, DataSheets
Output	• Describe the scope of output or system functions • Suggest how the output should be used for downstream tasks or user workflow	FactSheets, Model Cards

Figure 7: Mapping between user questions and possible XAI techniques by Lia et al. [11]

point between designer and AI engineers in their step-by step process for the question-driven design of XAI(Figure 8). Lia et al. tested their 4-Step-design-process with a multi-disciplinary team from IBM that developed a tool to predict a patient's risk of healthcare adverse events and presented their findings resulting from this use case as additional support.

The first step, *Question Elicitation*, aims to understand the user requirements. It tries to answer what questions users have for understanding the AI, what the motivations for these questions are and what a good answer would consist of. Ideally, this should be done through user interviews, but it can be also done heuristically by the designer.

In the second step the *questions are analysed*. Here the key user requirements should be used to find a prioritisation for the types of explanations to provide. For this, the questions elicited in step one should be clustered and prioritized, either based on the frequency or collaborative in the team.

The third step is to *map the question to the model solution*. For this, the mapping guide (Figure 7) can be used. It is also the transition point from designers to engineers.

The last step is the *iterative design and evaluation process* in which prototypes get de-

	Step 1 **Question Elicitation**	Step 2 **Question Analysis**	Step 3 **Mapping Questions to Modeling Solutions**	Step 4 **Iterative Design and Evaluation**
Goals	• Elicit user needs for explainability as questions • Gather user intentions and expectations for the questions	• Categorize elicited questions and identify priorities • Identify key user requirements for the XAI UX	• Map prioritized question categories to candidate XAI technical solutions as a set of functional elements that the design should cover	• Create a design including candidate solutions and evaluate it with the user requirements • Iteratively improve the design to close the gaps
Tasks	1. In user research, first define the problem the AI solves, the AI tasks and/or user journey 2. Elicit questions that the AI needs to answer in the tasks or user journey to be useful, effective and trustworthy 3. Also articulate the intentions behind asking the questions and expectations for the answers	1. Cluster similar questions across users into categories 2. Prioritize categories that the XAI UX should focus on. This can be done by ranking the quantity of questions or collaboratively with the team 3. Cluster and summarize user intentions and expectations to identify key user requirements for the XAI UX	Work with AI engineers to collaboratively identify candidate technical solutions to generate explanations for each prioritized question category. A mapping guide to available techniques in XAI toolkits is provided. Sometimes it requires implementing solutions specific to your model	1. Create an initial design with the set of candidate XAI solutions 2. Evaluate the design holistically with the user requirements identified in step 2, ideally with user feedback 3. Iteratively improve the design and evaluate with the user requirements to close the gaps 4. If applicable, define success metrics based on the user requirements to guide the improvement of models and XAI techniques
Stakeholders involved	Designers, users	Designers, optionally the product team	Designers, AI engineers	Designers, AI engineers, preferably users
Supplementary guidance	• When access to users is limited, designer can use XAI Question Bank as a checklist to guide the elicitation. Designer can also adapt it to create a customized checklist • The team can also use the checklist to heuristically identify applicable user questions • A complex AI system may need to be broken down into steps or scenarios to elicit questions separately • For highly novel systems, scenarios or low-fi prototypes can be used to elicit questions	• For supervised ML, XAI Question Bank can be used to guide the categorization into 9 categories. However, categorization can be flexible based on the questions elicited • The analysis offers an opportunity for the team to develop a shared vision, so it could be beneficial to involve the team to vote on the user questions as team priorities, for example by using a prioritization matrix	• Depending on the XAI expertise level of the team, additional training or educational materials may be needed before scheduling this step (suggestions in the supplementary material) • Depending on the product stage, this step could impact not only the choice of XAI techniques but also the model (e.g., a simpler, more explainable model)	• In addition to the mapping guide, AI engineers can provide additional exemplars of output of XAI techniques to support design ideation • When access to users is limited, user requirements can be used for a heuristic evaluation within the team • Design iterations often impact, and should be developed in parallel with modeling solutions • New questions, user requirements and additional XAI techniques can emerge and should be incorporated in future iterations

Figure 8: Design Process for creating XAI by Lia et al. [11]

veloped and tested. In the early stages, the evaluation can be done by the team based on the user requirements, later users should be involved in the testing.

This design framework will be used in the following to identify the most important explanations for a conversational movie recommender.

5 Question Elicitation and Analysis

5.1 The task, user and journey

In this work, a conversational movie recommender will be used. So we have generally a CRS in a low risque area that will be developed for everyday people with no AI background and diverse knowledge about movies. The main task is to recommend the user a movie to his liking and thereby solve the problem of information overload in the movie area. For this, the CRS will ask for the user preferences and then use a context recommender to provide the user with a movie that is closest related to his preferences. The user can then accept the recommendation, ask for a new one or critique it by updating his preferences. When the recommendation is inconsistent with the actual preferences of the user and the mental model the user has about the system then a need for explaining the decision arise. In the following a heuristic analysis will be conducted to determine the user needs, what kind of questions a user might ask and what answer they expect.

5.2 Framework for identifying user-centered questions

A first indication of what question is of special importance can be derived from the work of Ribera and Lapedrizia [15]. The authors developed an explainability framework in which the users are classified into three main groups and presented with a target-group-specific explanation. For this they classified XAI along four dimensions: Why (corresponding with the motivations in Section 2.1.1), How (corresponding with the taxonomies of methods in Section 2.1.2), What (corresponding with the question bank by Lia et al. Section 4) and How to evaluate. For the what they identified 6 questions from a small literature review, which is a subset of the questions identified by Lia et al. (in the following marked in parentheses).

- (1) What did the system do? (output)
- (2) Why did the system do P? (why)
- (3) Why did the system not do X? (why not)
- (4) What would the system do if Y happens? (what if)
- (5) How can I get the system to do Z, given the current context? (how to be that)
- (6) What information does the system contain? (data)

Afterwards, they identified three main groups that use XAI:

- Developer and AI researchers: creators of the AI systems.
- Domain experts: specialists in the area where the decisions are made (for example lawyers in legal tech).

- Lay users: recipients of the decisions.

and proposed that every explanation, and especially explanations for non-AI experts, must follow the cooperate principles of Grice:

- Quality: only provide high-quality explanations by not giving explanations you believe to be false or for which you do not have enough evidence.
- Quantity: Give enough information, but also not more than required.
- Relation: Only give relevant information, maximize the quantity.
- Manner: This part concerns not what the explanation should be but how the explanation should be delivered. It states that ambiguity and obscurity should be avoided and that the explanation should be brief and ordered.

Based on these findings they developed a user-centred XAI framework that maps answers to the question of Why explain, What to explain, How to explain, How to evaluate the explanation to the three user groups (Table 2). *Developers and AI researchers* are

	Developers	Domain experts	Lay users
Why	Verification, Testing, Improvement	Learning, Adoption	Compliance with legislation, increase of trust and use
What	Global model, Data representation, Why and Why not	Local explanations, Why and why not	why not
How	Intrinsic, Post-hoc	Post-hoc, Visualization, Natural Language	Post-hoc, Brief, Plain Language
Eval	Completeness test, Performance	Tests of Comprehension, Performance, Survey of trust	Satisfaction questionnaires

Table 2: In this table after Ribera and Lapedrizia [15] answers to the 4 questions of Why, What, How, Eval(evaluation methods) are being mapped to the 3 different kinds of user: Developers, Domain Experts and lay users

mostly interested in using explanations as a way to test, verify and improve the AI system they developed. They can understand explanations based on model internals of intrinsic models and complex data representation. For the testing of the model, they are most interested in why the system did P and in why the system did not do X. The global state of the model can also be of high interest for this group.

Domain experts ask questions to learn from the system and to better adapt their decisions. For this, they are most interested in answering locally why the system did P and not X. Such explanations should be created Post-hoc and be presented visually or by natural language, in the domain language of the user. The explanation should further be asked proactively by the user.
Lay users have either a right to an explanation by law or they get provided one so that the trust in and use of the system increases. For end-users counterfactuals are of particular interest, so answering the question of why the system did not do X can be especially helpful. The explanation should be created Post-hoc and delivered in brief and plain language.

So we can note that answering the question why not ... is of central importance. The answer to the counterfactual question should be brief and in plain language.

5.3 Counterfactual explanations

The work of Ribera and Lapedrizia give a clear indication that counterfactual explanation of the form why not, are central for satisfying the user's explanation needs. A central work in this area is from Miller [12]. In this work, the author derives from the social, psychological and philosophical science that human "explanations are contrastive [and] sought in response to particular counterfactual cases". This means every causal question is of the form: why P and not Q. With P being the recommended Item and Q the foil, that means the item that was not recommended. Even when people just ask why P there is an unexplicit foil. In normal human conversation, the foil is often inferred by the other person and while this might also be possible for a system with good knowledge about the user. The system could infer a foil from a similar movie of the recommendation history, it does not seem practical in a cold-start problem. Therefore the user should explicitly state his foil and if not stated the system should actively ask for one. Such counterfactual questions get asked when an abnormal event occurs, in this case when the system recommends a movie that does not match the real user preferences and the user's current mental model of the system. In such a case the user has a very good idea of what kind of movie they expected and can therefore ask explicitly why his expected movie was not recommended. Answering this question has several advantages. First, the user can cope with the inconsistency between his mental image of the system and the action of the system. The explanation serves here as a social interaction in which a shared meaning is created by updating the user's mental model of the system. This will help the user to restore his trust in the system and give him further insights into the inner working of the system. And secondly, the system can also answer the question of how to be that, because it not only provides the user with an explanation of why the foil was not recommended but also how it could be recommended. This will help the system to improve its performance, by updating the user's preferences to better reflects the real user preferences. It may also present the user with a suggestion for a category they have not used yet. Therefore helping the user to adapt usage and increase their

investment in the system.
Miller additionally identifies in his work how such a counterfactual explanation should be given. Stating that explanation should not represent the whole truth, but rather a small selection. This is consistent with the suggestion of Ribera and Lapedrizia for a brief explanation. For this Miller identifies several properties on which a counterfactual explanation could be evaluated and filtered.

- *Fact and foil have do be diffrent*: A cause that explains both fact and foil is no good explanation.
- *Social*: Because explanations are a social interaction between explainer and explainee, they should take the social context, environment and audience into account when making an explanation. This means that the movie sophistication and knowledge about actors and directors should be taken into account.
- *Intentional factors are more important than unintentional*: Causes that are intentional should be preferred over unintentional causes. Meaning that a preference that was set intentional has high importance.
- *Focus on the abnormal*: Abnormal causes are especially interesting to humans. This means that when an abnormal feature is influential to a prediction, it should be included in the explanation. Meaning that an uncommon factor should rather be used as an explanation than a common one. However, it is to note that a user should still be familiar with it.
- *Necessary causes are preferred to sufficient*: A cause that is necessary to achieve the foil is better than one that is just sufficient. However, in our case, no one fact is either fully necessary nor sufficient.
- *Responsibility*: A cause should have a high responsibility. This is closely related to necessity.
- *"Probabilities probably don't matter"*: Statistical probabilities are not as important as causal relationships. Statistics should never be used alone as an explanation.
- *Truthful*: An explanation should be true. However, truthfulness alone is only necessary but not sufficient.
- *Consistent with prior beliefs of the explainee*: People tend to ignore explanations that are inconsistent with their beliefs. Therefore the system should try to balance truthfulness with belief consistency so that enough new information is provided without antagonizing the beliefs of the explainee. This conflicts with the point that intentional factors are more important, suggesting that the number of explanations staying in conflict with set preferences should be limited.
- *General and Probable*: Explanations should explain a large number of events. This is in contrast with the focus on the abnormal and should therefore only be used when there is no abnormal case.

These criteria should later be used to create the counterfactual explanation.

5.4 Result explanation

Explaining the result, so in this case, the recommendation is for a recommender an evident necessity. Especially if the user does not know the recommended item beforehand it is central to provide the user with the necessary information to judge the recommendation. This kind of explanation is provided by almost all CRS systems and is often not recognized as an explanation. However such information can be categorised as an output explanation. Pecune et al. showed in their work about social explanation [14], the stating of movie facts is one of the most used forms of explanation in human conversation about explanations. Such kinds of explanations help the user to gain evidence about the recommendation and appropriately evaluates them. It can be argued that this type of explanation is the most important because returning only the title makes it almost impossible for the user to judge the recommendation. Providing the user with information about the movie might help to identify further preferences that he wants to add and thereby improve the recommendation process. It could furthermore help to increase the trust when the user identifies similarities between the movie and his preferences.

5.5 Context explanation

Based on the related work (Section 3), especially the work of convey by Jain et al. [6] and the work of Narsucci et al. [13] a clear need for stating the systems current beliefs can be derived. CRS have to deal with ambiguous statements by the user. Such statements can be easily misinterpreted by the CRS. Additionally, the user might forget what preferences they have stated early in the conversation. So in CRS, there arises a special need for users to ask the system what its beliefs about the user preferences. Such a question can be categorised in the data/input categories of Lia et al. or the question (6) what information does the system contain? in the framework of Ribera and Lapedrizia. However, it is important to note that while the developer of the framework had the training data side in mind when classifying this question, a user in a CRS system is more interested in his input side. So answering the question what is the current context, are your current beliefs about my preferences is a central question a CRS must be able to answer.

5.6 Other explanaitions

Besides the already named questions, the framework of Lia et al. provides 6 more questions categorise.

Why: Providing the user with causal why explanations is probably the most used form of explanation. They give, like counterfactual explanations, insight into the causal reasoning behind the system. However, as Miller clearly state "people do not ask why event P happened, but rather why event P happened instead of some event Q" [12]. So counterfactual explanations come more natural to humans, especially lay users. They are also not as effective in restoring a users mental model of a system after an unexpected recommendation. Answers to why questions are also not able to provide the user with new suggestions for his preferences and are therefore not as effective in improving the recommendation process. Overall simple causal why explanations are inferior to the counterfactual why not explanations.

What if: Providing the user with hypothetical questions, like what would happen if I add genre: action movie to my preferences, is not very helpful because the user can just do it. The user has full control over his own preferences and changing them is straightforward, so adding this form of hypothetical questioning does not provide any benefit.

How to still be this: This form is only a little help for a lay user in a low-risk area like movie recommendation. While such questions may provide the user with insights into the working of the system, and even provide additional preferences they could add or delete, they would probably not be used much. Such a form of deep understanding is only necessary for high risque areas and was even there not ranked high. However, studying such explanations might be a further research opportunity.

Performance: This is an easy to derive explanation. However, it only provides insight for people who knows how to interpret the metric. They are also not necessary and low ranked in low risque areas, as it is also stated by Lia et al. in which a wrong prediction has no big consequences. For a lay user, such explanations might even be counterproductive by confusing and unsettling the user. They may therefore even increase mental demand and decrease the trust in the system.

How: Describing how exactly the system makes its prediction is only necessary and useful for users with AI knowledge, especially for quality control. So clearly not in the case of CRS movie recommendation.

Other: There are no other important questions to answer neither on change, social impact nor terminology.

6 Implementation

In this chapter the implementation of the explainable conversational recommender system ABED[2] an Artificial Bot that can Explain its Decision will be described. This includes the bot, the recommender and the explanation corresponding with steps 3 and 4 of the framework by Lia et al..

6.1 The Bot

For the creation of the Bot, meaning the whole conversational part of the CRS the framework RASA[3] was used. Rasa provides an open-source framework to create machine learning powered bots. it uses python for customisation and yml for creating and saving training data. It also comes with support for pre-trained models, which minimize the amount of training data to get the bot running. It furthermore provides endpoints for several different chat applications, including mattermost and telegram, as well as the possibility to run and test it with RASA X[4]. The current bot uses rasa x to deployed the bot but changing it to one of the supported applications can be done with minimal effort.
For the *input processing* all intents (see Table 3) and entities (see Table 4) have to be declared in the domain.yml file.

Intent	Intent type after Table 1	Used entities
Greet	Initiate Conversation	None
Bot challenge	Not defined in Table 1	None
Add preferences	Provide Preferences	Movie attributes
Delete preferences	Revise Preferences	Movie attributes
Ask recommendation	Ask for recommendation	None
Context explanation	Obtain Explanation	None
Ask counterfactual explanation	Obtain Explanation	Movie
Ask result explanation	Obtain Details	Movie attribute name
Rate	Feedback	Rating
Help	Not defined in Table 1	None
Restart	Restart	None
Goodbye	Quit	None

Table 3: Implemented intents

For every intent a couple of examples have to be provided. These examples are defined in the nlu.yml file and have to specify the entities that should be extracted in the given

[2]https://github.com/benebjoern/ABED
[3]https://rasa.com
[4]https://rasa.com/rasa-x/

example. Rasa provides two major ways of intent classification; pretrained embedding and supervised embedding. Pretrained embedding uses the spaCY library[5] to load a pretrainde language model. This approach uses word embedding, in which similar words are represented by similar vectors. Because such a pretrained model already knows the similarity between words it requires less training data and trains also faster. For this reason the bot uses currently this approach. However the supervised embedding might be better suite when enough training data is gathered (at least 1000 examples per intent). The supervised embedding uses tensorFlow to train the language model from scratch. This comes with the advantage that it can better adopt to domain specific language. Furthermore it would allow to classify multiple intents in one message. This would be especially helpful for the add and delete preference intent. Both uses at the end the vectors and extracted entities to classify the intents using sklearn intent classifier. The entities get extracted through two components. First the RegexEntityExtractor

Entity	Description	Extraction support
Genres	Movie attribute: genre of the movie	Lookup table
Keywords	Movie attribute: keywords describing the subject	Lookup table
Actors	Movie attribute: cast of the movie	Lookup table
Directors	Movie attribute: directors in the movie	Lookup table
Year	Movie attribute: year of release	Regex function
Decade	Movie attribute: decade of release	Regex function
Time	Movie attribute: length of the film as category	Synonyms
Minutes	Movie attribute: length of the film in minutes	Regex function
Rating	Movie attribute: rating of the movie	Synonyms
Attribute name	Name of the above movie attributes	Synonyms
Rate	rating by the user either positive or negative	Synonyms

Table 4: Implemented entities

extraction, that uses Lookup table and Regex functions to identify entities. For this all actors, directors, genres, keywords and movie titles that the recommender knows are provided in Lookup tables. When one of these words appear in the user input it gets automatically extracted. This approach is very reliable, however it sometimes extracts an entity that was not meant, there is for example a movie that is named can and it can not cope with typing or spelling errors. For the extraction of year, decade and minutes regex functions are used. Regex functions are looking for specific patterns in the input, for example four digits for year entitys. The other aproche is the CRFEntityExtractor, that uses the training data two identify the entities. This approach is quite useful because it can also extract entities that are written wrong. A wrongly written entry can then be used in the back-end to find the most similar written entries and to ask the user for clarification. This is inpired by the work of Narsucci et al. [13] (see Section 3.2). While this approach currently is extracting only a fraction of all entities, it can be expected to

[5]https://spacy.io/

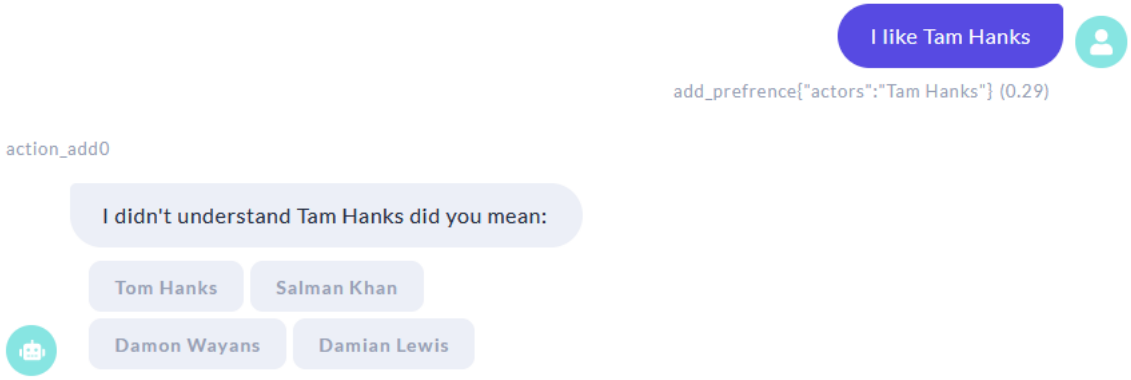

Figure 9: Interface asking clarification for wrongly written entity

improve over time, when more trainings data is added. Additionally both extractor can profit from the synonym tables which give synonyms for one entity, for example awesome is a synonyms for an excellent rated movie. For the movie categories length and release two entities can be extracted. This is due to the fact that release is transformed into decade categories to minimize the attribute vector. The same goes for length which is transformed into the time categories short, medium and long. Rating is also categorical with the possible values being excellent, good, okay, and bad (in more detail Section 6.2) Extracted entities get only saved for a short time, there are so to speak the short term memory of the bot. For long term memory slots have to be used. This bot has 4 slots:

- *Preferences* save the user preferences in a binary vector
- *Enough Information* is initiated false and gets set true when at least three preferences are stated
- *Recommended* is a list of all recommended items
- *Rated* a list of tuples with the first value being the rated item id and the second being the users rating (either positive or negative)

The *dialogue management* is not done through with rules and stories instead of a classical state machine. An ideal dialogue flow can be seen in Figure 10.

Rules are hardcoded instructions that are defined in the rules.yml file. Here the developer can specify what action should be invoked when specific intent is identified, or a specific chat has been invoked. In this case, it is mostly used to map specific intents to their actions. For example, when the user utters an add preferences intent, the corresponding action add_preferences should be invoked. It additionally can take set slots into account. For example, the bot has the enough Information slot which gets set to true if at least 3 preferences are set and changes to false if it gets under 3. When the slot is set to false the bot ask the user after adding or deleting for another preference of one of the movie attributes that was not set yet. If it is set to true it provides the user with

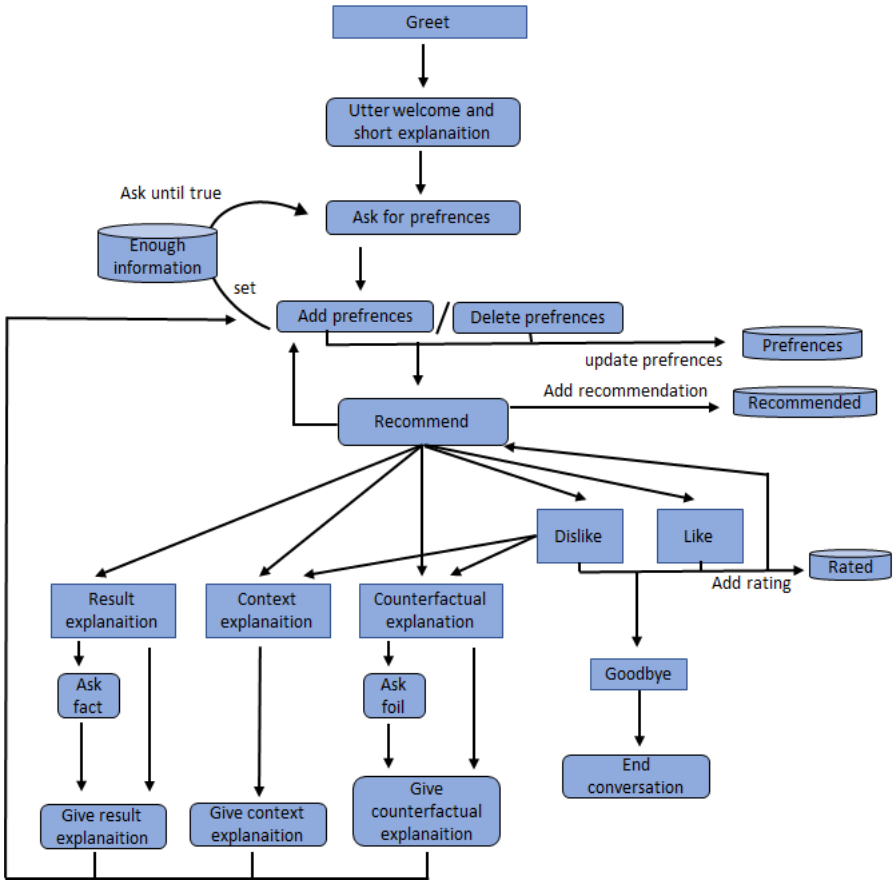

Figure 10: Dialog flow of the bot

a recommendation. The next way in which the dialogue can be managed is through stories, which are defined in the stories.yml file. Stories are more flexible than rules. They provide the Bot with exemplary dialogues, based on which the bot can decide the next action by matching the current chat history with the closest related story. Stories can thereby also manage wrong chat flow and provide the bot with multiple alternative ways to act.
The *Output* is another way in which the conversation flow gets managed. The Bot uses a hybrid approach in which text and buttons get combined. This can help to guide the user through the dialogue by providing them with a set of possible next steps. This is inspired by the work of Narsucci et al. [13] (see Section 3.2).

6.2 The Recommender

The CRS uses a context-recommender, more specific a vector space model (VSM) witch is defined in the recommender.py file. The recommender uses the movie attributes of

genres, keywords, actors, directors, length, release, rating to find the movie with the highest similarities between the user's preferences and the properties of the movie. The recommendation space gets represented by a matrix in which every row represents a movie and every column a specific property, for example, action movie (genres). If a movie has the property then the cell is true, otherwise false. The user preferences are represented by a vector which is first initiated with false values. Every time the user states a preference the corresponding cell gets set to true. For generating a recommendation the cosine-similarity between the user preferences vector and every row of the matrix gets computed. The result is a vector in which every value represents the similarity between one movie and the user preferences. In the last step, the highest n+1 values get sorted, with n being the number of already recommended movies, and the movie with the highest similarity that was not yet recommended gets chosen as a recommendation and added to the list of recommendations.

One problem with this approach is that the matrix can easily become very big, creating

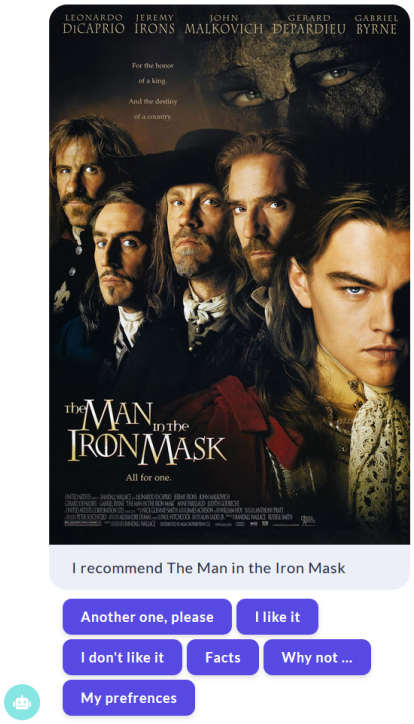

Figure 11: Interface for providing recommendation and next options

a challenge for both the memory and the computation demand. For this reason, the matrix is sized down, by both deleting uncommon movies and uncommon properties. Additionally, the release year was transformed into the decade and the movie length was divided into three equally sized categories short, medium and long. The rating was also categorised with the best 10% being categorised as excellent, the top third (excluding

the first 10%) as good, the medium third as okay and the bottoms third as bad. By this approach, the original data set [4] was downsized to (16136, 9100) meaning that the matrix represents 16,136 movies with 9,100 properties that can be critiqued. This transformation together with the creation of dictionaries and lists to map values to positions and vice versa was done in the data_preparation.ipynb file.
After recommending an item the system provides the user with several options. First asking for another recommendation, rating it positive or negative, or asking for an explanation.

6.3 Counterfactual Explanations

The use of a vector space model comes with the advantage of being intrinsically explainable. Therefore a self-programmed XAI method is used, instead of a post-hoc explanation method. Nonetheless as presented by Lia et al. in their question bank there are also predefined methods that would allow producing similar results for black-box recommender, especially CEM provides similar functionalities as the here created method. However, the use of a self-programmed solution allows to better customize the explanation to the requirements of a human-friendly explanation as defined by Miller (see Section 5.3). Miller describes the creation of a counterfactual explanation as a two-step process. In the first step, a complete explanation is generated and in the second step every part of the explanation is evaluated and a small amount of causes is filtered out. For the creation of the full explanation, all differences between the user preferences, the recommended movie properties and the foil movie properties get computed. Meaning that if a property is set in the foil it can not be set in both the recommended movie and the user preferences, such properties get added two a list of properties that have to be added to approach the foil as a recommendation. The same goes inverted for the case that the property is not set in the foil. Such properties get added to a delete list. This means that the first requirement, fact and foil have to be different is fulfilled. The result, a tuple of two lists gets then filtered based on several criteria. First properties that were set and would be needed to be deleted have a higher priority than properties that have to be added. This is due to the rule that intentional factors are more important than unintentional. However, there is a rule that an explanation should be consistent with the prior beliefs, therefore the amount of factors that are inconsistent with the preferences of the user is caped to half of all factors. Next, the values get ordered by their commonness, with uncommon features being ranked higher than common ones. This helps to fulfil both the criteria that one should focus on the abnormal, with uncommon features being more abnormal than common ones and increase the responsibility of the feature on the recommendation. Because uncommon features have a higher influence in increasing the likelihood that the foil gets recommended. However, choosing uncommon features comes with the danger of presenting the user with an unknown feature. Especially the knowledge about actors and directors is very diverse between different users and an explanation should take the knowledge of the recipient into account. Therefore the user knowledge about actors and directors gets calculated by identifying the least common

actor or director in the user's preferences. If an actor or director is more uncommon than the least common one in the user's preferences, it is assumed that the user is unfamiliar with it and it gets the lowest priority. In the end, the first 6 factors get chosen as an explanation and presented to the user.

The identification of the foil proposes a couple of difficulties, taking the movie 'Les

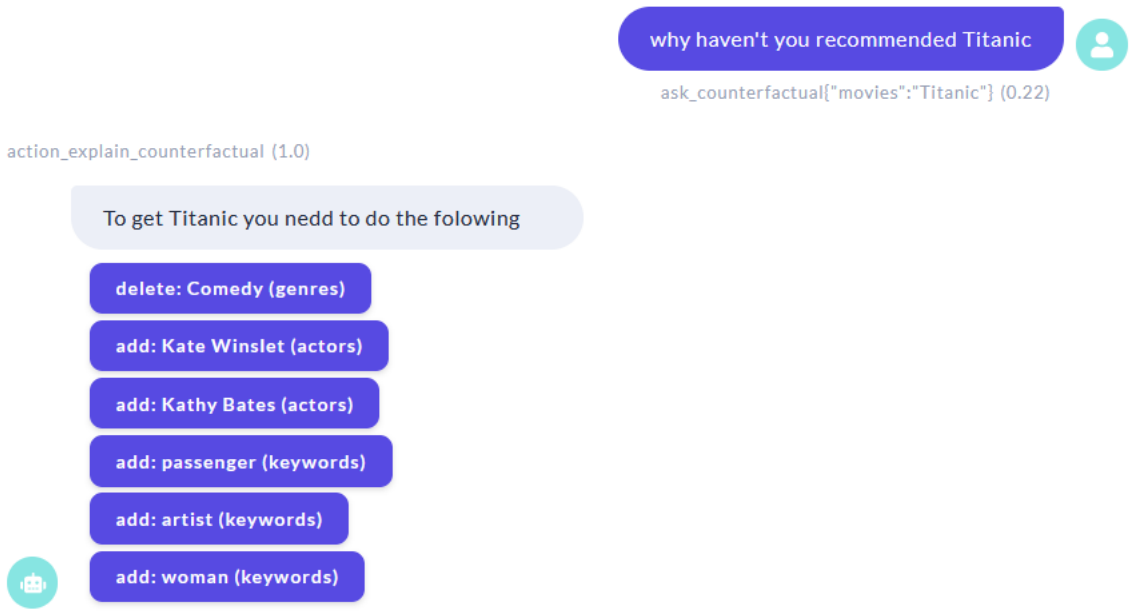

Figure 12: Interface for providing counterfactual explanation

Misérables' as an example. The title is quite complicated written and is likely to be not written correctly by at least some users. In such a case the system can look for the most similar written titles and ask the user to clarify. The other problem is that some movie titles are assigned to multiple movies. There are for example 6 movies with the title 'Les Misérables' in this data set. In cases where multiple movies are assigned to one title, the movie with the highest similarity to the user's preferences gets chosen as a foil.

6.4 Result Explanations

The user can ask for a result explanation, meaning details about the recommended movie. The user can ask about every possible movie attribute category and the plot. In the case of an attribute category that was transformed into more general categories like rating, the user gets information about both the exact value and the category in which it is thereby. For example, in the case of rating, the system might say that the movie has an IMDB rating of 9.7 which is categorised as excellent. In case that the user does not specify about what aspect of the movie he would like to receive more information, he gets asked to specify it. The result is always given in short and plain text. The

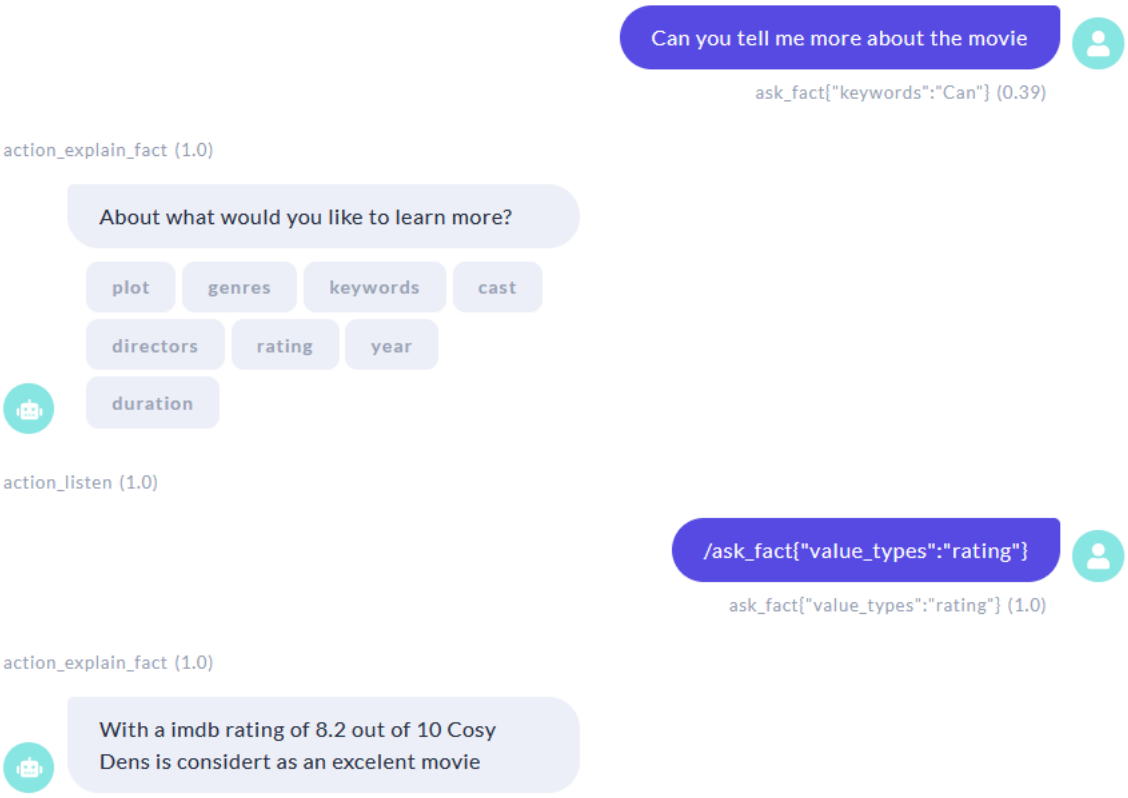

Figure 13: Interface for providing result explanation

explanation has to be asked actively, but there is a button after recommendation, that points out the possibility of this explanation.

6.5 Context Explanations

The user can ask the system at any point what its beliefs about the user preferences and it actively suggest this option after recommending a movie. The context explanation

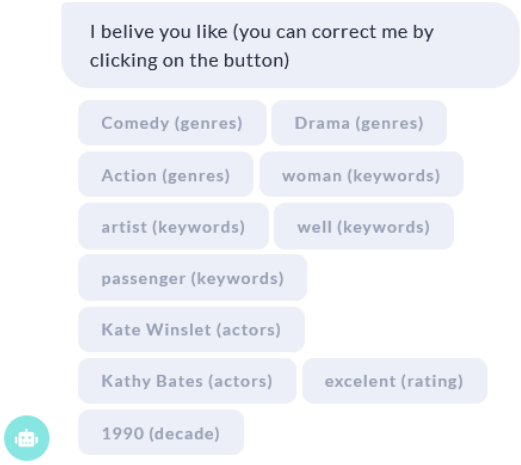

Figure 14: Interface for providing context explanation

provides the user with a complete list of all preferences the system beliefs the user to have. This list gets provided through buttons. This is for one a neat and simple way of providing the information and it additionally allows the user to easily correct the systems preference beliefs by just clicking the button to delete a wrong belief.

7 Discussion

The analysis of the user needs through the framework of Lia et al. has revealed both new and already known ways of explaining recommendations in CRS. The three identified explanations, result, context and counterfactual explanation can be ordered by the insight they provide.
Result explanations provide the user only with an indirect insight into the working of the bot. It only provides facts about the recommendation. The user can then based on this information try to derive the inner working of the bot. For example, he might see an attribute he has stated early and can derive that this attribute influenced the recommendation. However, such an explanation does not help the user in restoring his mental model when an unexpected recommendation was made.
Context explanations provide direct insight on what the beliefs of the system are. The big advantage of this explanation is that it provides a point of reconciling the believed preferences of the system with the actual preferences of the user. This is especially helpful when the user has forgotten what preferences he has stated or when the system has misunderstood a stated preference. However, a user still has to derive for himself how exactly these beliefs affect the recommendation. It only helps in situations in which the difference between the expected outcome and the real outcome is induced through a difference in the believed preferences and actual preferences.
Counterfactual explanations provide the user with a causal explanation. They thereby not only provide insight into the system's beliefs but also how this affects the recommendation process. The big advantage of this method is that it is most useful when an unexpected event occurs. Unexpected events damage the users model of the system and create thereby a high need for an explanation to restore this model. Counterfactual explanations are best equipped to bridge the difference between the users model of the system and the real state of the system. They do this by pointing out the difference between the, by the user proposed, foil and the recommendation of the system. This helps the user to update his model of the system. It additionally provides the user with the possibility to change the system to better represent his real preferences by answering the question of how to get the foil. This is also the big difference to the causal why explanation. While causal why explanation might also help to provide insights into the causal working of a system, they do not help the user to cope with unexpected events. A why explanation might be a good way to justify a recommendation that fits the mental model of the user. However, a simple why explanation is not very helpful in an unexpected case and managing these unexpected events is the most important task an explanation has to fulfil.
All explanations are only given at the request of the user. While the possibility of receiving them is indicated after a recommendation and again after disliking a recommendation, it is important to give the user the choice of when to receive it. This is since the need for an explanation defence on the unexpectedness of the recommendation. A recommendation that fits into a users mental model of the system does not require an explanation and giving one would only increase the mental demand. Context

and counterfactual explanations interestingly mirror the way a CRS works. The context explanation can be seen as a mirror of the preference elicitation phase of the CRS. The CRS system is asking the user in this phase about his preferences to create a better model of the user. The context explanation does the same but only reversed, the user asks the system about his beliefs to build a better mental model of the system.
Counterfactual explanations mirror the critiquing process in a CRS. When a CRS gives a recommendation a user will critique it based on his preferences, again helping the system to update his model of the user. In case of a counterfactual explanation, the user gives the system a suggestion and the system critiques it based on his beliefs about the user's preferences.
While this observation is interesting, it is not surprising. CRS is built to create an accurate model of the user, so it is only consequent that similar methods can help the user to build a model of the system.

8 Conclusion

The focus of this work is on the analysis of the user requirements for recommendation in CRS and the development of a basic XCRS system, that can be used in further research. It has created with Abed an XCRS that is based on a broad heuristic study of users needs in low-stake recommendation areas for end users. This bot can be used in further studies to advance the research in the field of XCRS.

8.1 Limitations

The here presented work comes with some limitations.
First, the requirements for the bot were chosen solely heuristically. However, the question elicitation and analysis is based on broad research and includes work with user studies as well as extensive literature revise.
Another limitation is that the bot was not tested by users. The only contact it had with users so far was for improving the natural language understanding (nlu) model. It, therefore, has only partially completed the last step of the Lia et al. framework by heuristically evaluating the implemented explanations (Section 7). However, the lack of clear user feedback makes it rather a prototype than a finished end-product.
While the nlu model was already trained with external input of real users it is still based on a comparatively low amount of training data. The nlu model still does some wrong classification and is not working perfectly.

8.2 Future Work

The next important step would be to present the bot to the end-user. This should help to further improve the nlu and to gather user feedback. The use of user feedback can be used for two purposes. First to iterate over the design process. The current design poses still a lot of questions that are not answered yet. For example the amount of information that should be presented and how it should be represented. Explanations in CRS come with their unique requirements for HCI and this field is still under-researched. The other important task will be to quantify the benefits of the here presented explanation and to compare them with systems with no or other types of explanation.

References

[1] D. V. Carvalho, E. M. Pereira, and J. S. Cardoso. Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8), 2019.

[2] Z. Chen, X. Wang, X. Xie, M. Parsana, A. Soni, X. Ao, and E. Chen. Towards explainable conversational recommendation. Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-2), 07 2020.

[3] P. Grasch, A. Felfernig, and F. Reinfrank. Recomment: Towards critiquing-based recommendation with speech interaction. RecSys 2013 - Proceedings of the 7th ACM Conference on Recommender Systems, 10 2013.

[4] J. Habib, S. Zhang, and K. Balog. Iai moviebot: A conversational movie recommender system. *Proceedings of the 29th ACM International Conference on Information amp; Knowledge Management*, page 3405–3408, 2020.

[5] A. Iovine, F. Narducci, and G. Semeraro. Conversational recommender systems and natural language: A study through the converse framework. *Decision Support Systems*, 131:113250, 04 2020.

[6] M. Jain, R. Kota, P. Kumar, and S. N. Patel. Convey: Exploring the use of a context view for chatbots. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, page 1–6, 2018.

[7] D. Jannach, A. Manzoor, W. Cai, and L. Chen. A survey on conversational recommender systems. *ACM Comput. Surv.*, 54(5), May 2021.

[8] Y. Jin, W. Cai, L. Chen, N. N. Htun, and K. Verbert. Musicbot: Evaluating critiquing-based music recommenders with conversational interaction. CIKM '19, page 951–960. Association for Computing Machinery, 2019.

[9] M. Lent, W. Fisher, and M. Mancuso. An explainable artificial intelligence system for small-unit tactical behavior. American Association for Artificial Intelligence, 01 2004.

[10] Q. V. Liao, D. Gruen, and S. Miller. Questioning the ai: Informing design practices for explainable ai user experiences. *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, page 1–15, 2020.

[11] Q. V. Liao, M. Pribić, J. Han, S. Miller, and D. Sow. Question-driven design process for explainable ai user experiences. ACM, 2021.

[12] T. Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38, 2019.

[13] F. Narducci, M. de Gemmis, P. Lops, and G. Semeraro. Improving the user experience with a conversational recommender system. *XVIIth International Conference of the Italian Association for Artificial Intelligence*, pages 528–538, 11 2018.

[14] F. Pecune, S. Murali, V. Tsai, Y. Matsuyama, and J. Cassell. A model of social explanations for a conversational movie recommendation system. HAI '19, page 135–143. Association for Computing Machinery, 2019.

[15] M. Ribera Turró and A. Lapedriza. Can we do better explanations? a proposal of user-centered explainable ai. *IUI Workshops '19*, 03 2019.

[16] E. Rich. User modeling via stereotypes. *Cognitive Science*, 3(4):329–354, 1979.

[17] M. Robnik-Sikonja and M. Bohanec. Perturbation-based explanations of prediction models. *Human and Machine Learning*, pages 159–175, 06 2018.

[18] R. Wexler. When a computer program keeps you in jail: How computers are harming criminal justice. New York Times, 06 2017.