Unpacking Human-AI interactions: From interaction primitives to a design space

KONSTANTINOS TSIAKAS and DAVE MURRAY-RUST, Delft University of Technology, The Netherlands

This paper aims to develop a semi-formal design space for Human-AI interactions, by building a set of interaction primitives which specify the communication between users and AI systems during their interaction. We show how these primitives can be combined into a set of interaction patterns which can provide an abstract specification for exchanging messages between humans and AI/ML models to carry out purposeful interactions. The motivation behind this is twofold: firstly, to provide a compact generalisation of existing practices, that highlights the similarities and differences between systems in terms of their interaction behaviours; and secondly, to support the creation of new systems, in particular by opening the space of possibilities for interactions with models. We present a short literature review on frameworks, guidelines and taxonomies related to the design and implementation of HAI interactions, including human-in-the-loop, explainable AI, as well as hybrid intelligence and collaborative learning approaches. From the literature review, we define a vocabulary for describing information exchanges in terms of providing and requesting particular model-specific data types. Based on this vocabulary, a message passing model for interactions between humans and models is presented, which we demonstrate can account for existing systems and approaches. Finally, we build this into design patterns as mid-level constructs that capture common interactional structures. We discuss how this approach can be used towards a design space for Human-AI interactions that creates new possibilities for designs as well as keeping track of implementation issues and concerns.

CCS Concepts: • **Human-centered computing** → **Interaction design**; *Interaction paradigms*; • **Computing methodologies** → *Artificial intelligence.*

Additional Key Words and Phrases: Human-AI interaction, design patterns, explainable AI, human-in-the-loop, hybrid intelligence

ACM Reference Format:
Konstantinos Tsiakas and Dave Murray-Rust. 2023. Unpacking Human-AI interactions: From interaction primitives to a design space. 1, 1 (January 2023), 46 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Artificial Intelligence and Machine Learning (AI/ML) models are in the midst of a transition from use in back-end data science systems, operated and interacted with by experts, to end-user focused applications that prioritise ease of use and interactional fluidity [77]. This transition to human-centered AI approaches is driven by several factors – the increasing power of the models and systems; the need for more engagement with real-world data; the enlisting of users as participants in model development through annotation or other feedback; the desire by more organisations to make use of the possibilities of AI/ML; and the corresponding need to make sure that the models are operating correctly or adapt them for particular tasks. This has given rise to a growing set of human-AI interaction paradigms, in particular human-in-the-loop (HITL) [15, 54], explainable AI (XAI) [42, 63] and hybrid intelligence (HI) and collaborative learning systems [20, 75]. This expansion in interactivity, coupled with the need to understand how systems grow and change

Authors' address: Konstantinos Tsiakas, k.tsiakas@tudelft.nl; Dave Murray-Rust, d.s.murray-rust@tudelft.nl, Delft University of Technology, Delft, The Netherlands.

Manuscript submitted to ACM

over time [30] and affect diverse stakeholders [13] leads to a need for new ways to design interactions, as we transition from *Human-Computer* to *Human-AI* interactions [78].

Human-AI (HAI) interactions can involve humans in different parts of the algorithmic operation, offering possibilities for designing new types of interactions, considering the different paradigms. XAI methods are used to provide additional information about the underlying AI processes [8] through communicating descriptions of model functioning [e.g. 9] even if it is not always a silver bullet [25, 57]. HITL approaches aim to enhance system's decisions by incorporating human decision making, when needed, to enable human oversight [33]. Human feedback can be integrated to the learning mechanism of ML models through interactive ML (iML) methods to facilitate model training, as well as to explore new forms of interactivity between humans and ML algorithms [50]. Hybrid intelligence and collaborative learning methods can combine human capabilities with AI, allowing both parts to efficiently communicate and exchange information in order to mutually make decisions, inform and learn from each other [4, 85].

Despite the computational advances and capabilities of AI models, designing HAI interactions remains a challenging task. Designers often have a broad understanding of the possibilities of AI/ML, but lack specifics [23]. This is related to two AI attributes: *capability uncertainty*, which indicates the uncertainty of what the possibilities of an ML model are, and *output complexity*, which covers the general difficulty of working with multidimensional, rich outputs from large systems [82]. Envisioning new AI-based solutions for a given UX problem should consider that HAI interactions need to adjust to different users and evolve over time [30]. Going beyond the one-to-one interactions with systems, there is a also the need to investigate the effects of automated algorithmic systems on on the different groups and types of users involved in or affected by the AI decisions [13], such as in organisational decision making [10]. Human oversight is an important aspect which enables users to maintain human agency and accountability, by participating in the decision making. From such a sociotechical perspective, hybrid (or shared) decision making can involve both decision-makers and decision-subjects towards interactions with contestable AI models [6]. However, designing for hybrid decision making can be challenging while considering the legal, social, technical and organizational issues [27]. Moreover, designing AI-based systems that facilitate *meaningful human control* requires the implementation of moral decision making methods in order to address responsibility gaps related to hybrid decisions [14].

Responses to this complexity and the need for human-centered AI design takes several forms, from high level frameworks through to new interaction paradigms and implementation methods. High level frameworks shape the way that people approach creating HAI systems. Shneiderman's *Human-Centered Artificial Intelligence* (HCAI) framework looks to create reliable, safe and trustworthy HAI interactions through active participation [64], with the intent to achieve a high level of human control, while maintaining a high level of computer automation. Xu's HAI framework [77] sets out three key components for system design: *technology enhancement*, *ethically aligned design*, and *human factors design*, highlighting the need to design responsible and reliable AI-based solutions. Yurrita et. al's multi-stakeholder framework looks at how human values connect to properties of AI/ML systems [84]. To address sociotechnical questions, new human-machine configurations use HITL techniques to augment the algorithmic system through model auditing and altering [33]. Such interactions enable the user to be part of the decision making process by querying, evaluating, and editing the underlying model and data. Similarly, a Human-Centered XAI approach (HCXAI) proposes to put the user and human factors in the center of technology design, taking into consideration the interplay between values, interpersonal dynamics, and the socially situated nature of AI systems [26].

Design guidelines and frameworks tend to work upwards from an interactional perspective, to support practitioners through a set of design methods, practices and examples for designing with AI, e.g., Google's People + AI Guidebook[1]. Other approaches focus on specific contexts and application areas to identify the most important design aspects and challenges for a given context, e.g., design guidelines for AI-based Tutoring Robots [79]. Similarly, design frameworks aim to address challenges related to *Responsible AI* by integrating ethical analysis into engineering practice [58], or by addressing ethical challenges for specific AI application contexts, e.g., ethical AI in K-12 education [86]. Research through Design (RtD) has been proposed as an approach to ensure that the role of AI in a system is legible to the end users [44]. Sketching and prototyping are RtD activities which could support the ideation and implementation of HAI interactions. At a higher conceptual level, metaphors can affect expectations of performance [38], help designers to understand key concepts [22, 52], or lead to envisioning new kinds of relations between humans and technology [45].

This work attempts to address the gap of ways to specify interactions between humans and models. Our proposed design space aims to be more pragmatic than high level frameworks, as it works from concrete actions at a user-model level, providing a link from guidelines and suggestions for design practices to the actual implementation and prototyping of existing and new types of interaction. It attempts to give designers and developers more direct facility to engage with the potentials for interaction with models by providing a palette of possibilities that map understandable human concepts to specific exchanges of information. We do this through proposing a communication protocol which can describe the interactions that take place around human user(s) and AI/ML model(s). We draw inspiration from agent communication protocols [e.g. 29, 56] which use communicative acts to enable agents to communicate their intent for a specific service [2, 3]. Working from this, the intuition behind this paper is that a common set of communicative acts can be defined to describe HAI interaction patterns as information exchanges whether by providing training examples to a model, validating a model's prediction or providing explanations. Such a specification would help to imagine richer interactions with models, and could allow a broader range of people to take part in the design of HAI interactions.

Our approach defines interactions between users and models based on the intent and type of the exchanged information. Towards this, we review existing frameworks and guidelines for HAI interactions, as well as application examples, in order to identify a common set of communication channels between users and AI models. Following our approach, we unpack existing HAI interaction into low-level communicative acts. Based on this, we define a set of HAI interaction primitives which consider both design and implementation aspects of an interaction (Figure 1). Our motivation for defining a design space based on HAI interaction primitives is to: (a) provide a space in which existing interaction concepts and paradigms can be represented, bridging the gap between human and machine understanding of what an interaction entails, (b) allow for the potential invention of new kinds of interactions through exploring the space of communicative actions and interaction patterns, and (c) provide a specification which carries the required information needed for the prototyping and implementation of HAI interactions. The key contributions of the paper are:

- A review of HAI interaction paradigms, design frameworks and guidelines for XAI, HITL, and HI, along with a set of use cases, focusing on the different types of HAI communication.
- A system of HAI primitives and types that can be used to encode the interactions between users and AI models in the form of actions and message passing, capturing the intent and type of the communicated information.
- A collection of interaction patterns as sequences of messages which can be used as design patterns for human-AI interactions to (a) reflect existing practice and (b) create new forms of interactions.

[1] https://pair.withgoogle.com/guidebook

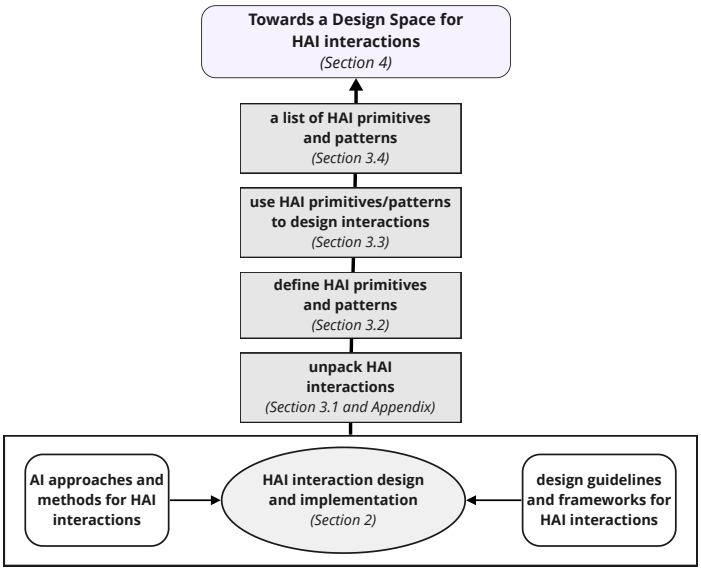

Fig. 1. Proposed Approach. We review existing guidelines, taxonomies and frameworks related to the design and implementation of human-AI interactions. Our goal is to unpack HAI interactions into a set of interaction primitives and patterns, based on which more complex interactions can be defined. Taking into consideration both technical and design challenges, we describe how the proposed primitives can be used to define a design space which can inform both design and implementation choices for HAI interactions.

The paper is structured as follows:

- In order to define the proposed communicative framework, we review existing taxonomies, frameworks, and guidelines for HAI interactions, including HI, XAI, HITL, interactive ML (iML), and collaborative learning systems (Section 2). Considering the range of these interaction paradigms, approaches and concepts, our goal is to specify the interactions between humans and AI models, based on the intent and type of the communication.
- From this, we extract a set of communicative acts that can be combined into interaction patterns to represent a range of HAI interactions within these dominant paradigms (Section 3). We build these acts and patterns from a combination of simple primitives and a set of types into human-readable verbs that specify an interaction. We demonstrate our approach by describing interactions from existing use cases and frameworks (Section 4).
- We discuss how a design space using the proposed HAI interaction primitives and patterns can be developed to support both designers and AI practitioners to design and implement HAI interactions considering both design and technical aspects (Section 5).

2 BACKGROUND AND RELATED WORK

The goal of the review is to identify the ways of communication between human users and AI models and characterize them in terms of the intent and type of the exchanged information, exploring existing HAI interaction paradigms. We provide an overview of design guidelines and taxonomies which provide suggestions on how to apply design principles for HAI interaction systems, as well as frameworks for design patterns. Finally, we present a set of HAI

interaction systems focusing on the design and implementation aspects, as well as approaches for communicative protocols, towards defining a set of communicative acts and patterns for HAI interactions.

2.1 Human-AI Interaction Paradigms

Human-AI interaction is defined as *"the completion of a user's task with the help of AI support, which may manifest itself in non-intermittent scenarios"* [71]. Following this definition, there are three main HAI interaction paradigms, *intermittent*, and *continuous*, and *proactive*, taking into consideration *"how differences in initiation and control result in diverging user needs"*. Intermittent HAI interactions are user-initiated and turn-taking interactions where the system provides a response for an explicit and predefined cue. Continuous HAI interactions utilize user's implicit input, as part of a continuous input stream, and provide a response which can either be accepted or ignored by the user. Finally, proactive HAI interactions are AI-initiated and triggered by predefined changes in the system. Since such interaction paradigms can exist in parallel, there is an increasing need to consider the open challenges while designing continuous and proactive interactions in terms of how users and AI systems interact with each other.

One of the key concerns while designing HAI interactions is the black box nature of AI models, where decisions and underlying operations are not visible or explained to humans. Moreover, human intentions are not always clear or they differ from the actual communicated action. This problem is known as the *"two black boxes problem"*, based on which both human cognition (cognitive intelligence) and AI are considered as black boxes. In order to address this problem, a symmetric and collaborative model for HAI interactions has been proposed, highlighting the need for explainability for both communication channels [74]. Based on this framework, Semantic Interaction (SI) is used as a design philosophy for symmetric and collaborative interactions between humans and AI. In the context of SI, XAI serves as a method to communicate information from AI to human users, while *explainable cognitive intelligence* (XCI) is used to provide information from humans to AI systems. A main challenge while considering both communication channels for the design of HAI is the *semantic gap*. Human high-level decisions and actions need to be translated to a set of model-understandable parameters, values and operations, while AI models must communicate its internal processes and decisions in a human-understandable way. HAI interactions should enable both AI systems and human users to efficiently communicate to each other and make decisions in a collaborative manner to augment both human and artificial intelligence, towards Hybrid Intelligence (HI) interactions. Akata et al. [4] unpack the research challenge of building HI systems into four research themes: *Collaborative*, *Adaptive*, *Responsible*, and *Explainable HI*. An open challenge towards designing interactions with HI systems is to develop methods for designing negotiation, agreements, planning, and delegation interactions in hybrid teams, considering the needs and role of the team members. *Usable* and *Useful AI* are two terms which describe how interactions with AI models can be designed to provide usable solutions that are easy to understand and apply in order to satisfy user needs [77]. Towards this, AI designers and practitioners should consider both the development of AI/ML models and the design of the interactions and behaviors around such models, including user-centric techniques, such as explainability and user control.

The integration of explainability and user control features creates new types of interactions between users and models. Designing interactions with (X)AI models should follow specific design principles towards the development of *Responsible AI* systems, related to fairness, transparency, and privacy aspects [8]. In terms of user control and feedback, HITL and iML techniques enable users to interact with AI/ML models in order to guide, steer and facilitate the learning process. Human users can be involved in the different stages of the ML pipeline: data extraction, integration and cleaning, iterative labeling, as well as model training and inference [15, 76]. Related to how human users can intervene with models, model contestation is defined as an interaction with "humans challenging machine predictions" [36]. A

proposed framework for contestable AI by design [6] highlights a set of sociotechnical features and practices for model contestation and hybrid decision making, including interactive controls, explanations, and intervention requests. Such approaches highlight the need for designing new types of interactions, considering the different interaction paradigms, leading to the development of design frameworks and guidelines for HAI interactions.

2.2 Design Guidelines, Frameworks and Taxonomies for Human-AI interactions

Design guidelines and frameworks aim to support designers on how to apply specific design principles while designing interactions with AI systems. Microsoft Research has proposed a set of guidelines for designing HAI interactions [7]. These guidelines provide design choices and examples for interactions between users and AI systems, considering aspects of system transparency, explainability, user feedback and control. Designing interactions which support such features requires human users to interact with the system at a *user-model* level. That means that designers should take into consideration the different ways that users can efficiently interact with an AI model, as well as the underlying design and technical challenges while prototyping such interactions. A process model for co-creation of AI experiences (AIX) describes how designers can be familiarized with AI models and their functionalities in order to include them in the design process as design materials [66]. However, there is a lack of design innovation in envisioning how ML/AI might improve and create UX value to users. Yang et al. [81] highlight the need to support UX practitioners by creating design patterns. They propose four channels based on which AI/ML capabilities can create UX value to users, namely *self*, *context*, *optimal*, and *utility-capability*, considering what users can infer about these aspects through their interaction with models. Such frameworks and guidelines aim to support the ideation process of designers while prototyping and designing interactions between users and an AI system.

Taxonomies and frameworks for XAI approaches and methods can support the design and implementation of interactions with explainable systems, considering different aspects of XAI-based interactions. A collection of XAI-based questions has been proposed as a design space for XAI interactions considering diverging user characteristics, e.g., user needs, role, expertise, and experience [41, 42]. Other taxonomies focus on the different aspects of the use case, e.g., problem definition, explanator properties, and evaluation metrics [63] or the relations between the use case, the AI system, and the explanation algorithm [1]. Design frameworks can provide guidelines on how to design and evaluate XAI-based interactions by mapping design goals to evaluation methods based on the target population [49]. Focusing on user evaluation for XAI interactions, Chromik et al. [18] classify evaluation methods based on task-related, participant-related and study design-related dimensions. Such frameworks can support the selection of appropriate XAI methods for a given interaction concept. Sperrle et al. [65] proposed a dependency model for XAI processes as a design space for human-XAI interactions, considering the potential bias that can be propagated during the interactions between different stakeholders and end-users. Wang et al. [72] followed a theory-driven approach to link explanation features to user reasoning goals, resulting to a conceptual framework which can inform the selection of appropriate explanations towards mitigating possible cognitive biases. Designing XAI interfaces should also consider specific design principles based on the interaction concept and explanatory goals [17]. Human users may interact with XAI interfaces for *information transmission*, where the goal of XAI is to help the user understand the underlying AI behavior, while XAI-interaction as *dialogue* refers to an iterative user-driven communication through user queries and AI responses/explanations. For both concepts, the AI behavior does not change in contrast to XAI-interaction as *control*, where the user provides feedback to adjust the (explainable) model until a desired AI behavior is reached, which relates to HITL/iML interactions.

HITL/iML methods enable human users to participate in the decision making and model training process. Design principles for user control and feedback interfaces include interactivity to promote rich interactions, providing explicit and clear task goals, support user understanding and engagement, and capture user's intent based on their input [24, 54]. Moreover, designing interactions with HITL/iML interfaces requires appropriate learning methods, considering new forms of relationships between humans and ML algorithms [19, 50]. The design and implementation of iML/HITL interfaces depend on the type of user feedback and can affect both user experience and system/model performance. Michael et al. [48] discuss how the different types of cognitive feedback can be integrated to the iML mechanism, i.e., *self-reporting*, *implicit feedback*, and *modeled feedback*, through different feedback mechanisms. For example, domain-expert feedback can be communicated to the system and translated to model updates, i.e., modify dataset or model parameters [16]. Such HITL/iML approaches consider the effects of the interaction on human engagement and feedback quality. Since explanations can be used to enhance user's understanding of the model's performance, they can play an important role in ensuring a high quality of user feedback. Explanations can be combined with interactive capabilities enabling users to train ML models from scratch, resulting to a closed loop of XAI- and HITL-based interactions [67].

In the domain of hybrid intelligence, a taxonomy for design knowledge organizes the design decisions for HI interactions, including XAI and HITL methods, along four dimensions: task characteristics, learning paradigm, human-AI interaction, and AI-human interaction [20].

Focusing on hybrid decision making, a common interaction paradigm is *backward reasoning* design where the system's goal is to provide correct outputs and explain them to the user, following an AI-driven interaction. *Forward reasoning* can provide more agency to human users, involving both human and AI in the decision loop in order to address the issue of system uncertainty [85].

Such guidelines, frameworks and taxonomies can help AI designers to select appropriate types and methods for HAI interactions, based on the system requirements and interaction goals. Designing interactions which comply with high-level guidelines can be challenging, especially when multiple requirements need to be met. Design patterns can be used to structure the interaction by defining smaller parts and combining them into more complex interactions.

2.3 Design Patterns for Human-AI Interactions

Describing an HAI interaction using design patterns is not straightforward but it can simplify the design of complex AI systems and make them transparent in terms of their implementation requirements. Focusing on co-creation applications with Generative Adversarial Networks (GANs), Grabe at al. provide a list of HAI interaction patterns which categorize interactions based on the AI's co-creativity support level [32]. These interaction patterns are sequences of actions which describe an activity, e.g. initialize, create, adapt model, and can be combined to design interactions with different co-creativity support levels. A co-creative framework for interaction design analyzes the interactions between users, AI models, and the shared product [60], resulting to a set of design choices. The design choices can define the behavior of the AI, as a generative, improvisational, or advisor agent. Design patterns in HAI co-learning are used to design interactions which enable humans and AI to share knowledge and experience. Learning Design Patterns (LDP) refer to sequences of interactions that aim to initiate and facilitate the co-learning process [62], either by identifying knowledge gaps that team members may have or by enabling team partners to learn from other team members. In the context of AI-based game design, AI can serve different roles based on the selection of the interaction pattern [68]. For example, AI can act as *role-model*, where users need to imitate an AI agent to complete a game, or as an (AI trainee) which enables the user to teach the agent to do something in the game.

Design patterns have also been proposed considering specific interaction paradigms. Focusing on collaborative learning and hybrid intelligence systems, humans and computers can learn from each other through an iterative process, combining human-in-the-loop with computer-in-the-loop interactions [75]. More specifically, users and computers can communicate through a set of learning process patterns: (a) *decision support*, (b) *exploration*, and (c) *integration*. Each pattern refers to different types of interactions between human users and computers as an exchange of inputs, outputs, and feedback/explanations. iML methods have been used as design materials for movement interaction design [31], as an approach to design interactions with models based on human movements. The interaction is defined as a closed loop between designers and software which allows designers to provide information in the form of movement examples and parameters, and receive AI test outputs and visualizations. Through this loop, both AI and designers can learn from each other; designers can reflect on how movement parameters may affect AI behavior and AI can update its internal models based on the feedback received from the designer. However, even prototyping such interactions would require a lower-level specification of how users and models communicate and exchange information during such interactions at a user-model level.

Design patterns have been used to specify and categorize the interactions with hybrid intelligence and reasoning systems at a user-model level [70]. A modular approach is used to specify patterns of interactions of hybrid systems, including both data-driven (ML) and knowledge-driven (symbolic AI) approaches. Based on this approach, data (numbers, texts, tensors, and streams) and symbols (labels, relation, traces) are the two types of instances that can be used for AI model operations. The authors present a list of design patterns for hybrid AI systems and demonstrate how such patterns can be used to describe interactions from existing applications. Their proposed approach specifies the interactions in terms of model operations, including training, inference, and transformation, and can be extended to capture the concept of different types of human actors included in interactions with hybrid AI systems.

2.4 Design and Implementation of Human-AI interactions

In order to explore how users and AI models communicate and exchange information during HAI interactions, we review applications of HAI systems, focusing on the interaction design and implementation aspects. Advances in computational and learning methods, as well as the plethora of human-generated data has recently led to innovative AI systems which can be used by non-expert users for complex tasks. More specifically, OpenAI has released two types of deep learning models which generate new content based on user's input (promtps); DALL-E generates images based on a textual description [59] and chatGPT is a Large Language Model (LMM) which generates structured textual content based on user's prompts and questions[2]. Despite their complex architecture and advanced learning methods, interactions with such models are straightforward; the user provides a prompt and the model returns the generated data. However, the implementation of the training process requires methods to integrate human feedback and expertise in order to facilitate model learning. The training process utilizes an HITL method to integrate human users to the learning process. More specifically, Reinforcement Learning from Human Feedback (RLHF) is used to integrate human expertise to the learning process. Based on this approach, human users provide different types of feedback based on the learning process step. During model initialization, human users demonstrate the desired optimal behavior of the model (response to a prompt). For model optimization, a reward model was trained based on user ranking of possible model outcomes (output) for a given prompt (input). Such an approach demonstrates how different types of users can interact

[2]https://openai.com/blog/chatgpt/

with different types of models, considering both the goal of such interactions, e.g., interface design, and their technical implications, e.g., selection of learning/update mechanisms.

Interactions can become complex when designing for explainability and user feedback, even with less advanced or pre-trained models. Designing an XAI-based interaction can serve as a channel to communicate additional information about the agent's performance for a given concept. For example, model transparency (e.g., visualizing model's confidence) has shown to improve user's trust during an interaction with a virtual agent with speech recognition capabilities [73]. More specifically, human participants interacted with a virtual agent and an underlying speech recognition system. The virtual agent was used as a visualization means for XAI-based feedback to enhance user's understanding and trust of the speech recognition system. During these interactions, the user provides a model input (utterance) and the model communicates both its output and additional feedback (prediction/explanation). HITL-based interactions can allow for user feedback and control within an interaction in various ways [19]. The type of human feedback (as well as amount, frequency, granularity, etc.) should be inline both with the user's characteristics (e.g., role, expertise, preferences, etc.) and the model's characteristics (model architecture, learning/update rules, etc.).

HAI interactions can also involve multiple users with different roles, expertise and intentions. In such interactions, an AI model should be able to interact with multiple users and behave in a different way based on the user's characteristics. An interactive RL-based framework has been proposed for a personalized robot-based cognitive game which involves two different types of users [69]. A player (primary user) interacts with a robot (RL) through a game-based interaction (robot sets a game difficulty and user plays the game) and a supervisor (secondary user) who can remotely supervise and control the interactions. In terms of communicated information, the player provides implicit feedback to the model through task performance and engagement, while a supervisor can monitor the model's decisions through a informative UI and modify the robot's decisions, when needed. Both communication channels contribute to the model's learning updates and the decision making process in a different way. Similarly, the Human Experience Transfer Model (HETM) [46] combines two interaction loops; one for a human expert (trainer) who guides the model learning process and one for the human learner (trainee) who can provide feedback during the learning interaction. The trainer loop aims to facilitate the model training while the learner loop aims to personalize the learner's interaction. These loops specify two different interactions with different communication channels and model update methods. Such approaches for designing and implementing HAI interactions highlight the need to specify design patterns considering the underlying implementation requirements. Our work moves towards a design space which takes into consideration the technical aspects while designing such human-model communications.

2.5 Communicative Protocols

Based on the Semantic Interaction paradigm [74], both types of interacting agents, humans and models/systems, should exchange information in a mutually understandable way, to ensure meaningful communicative acts from both sides. In order to do so, user actions and intentions should be translated into an understandable model-specific format. On the other hand, model actions should be effectively communicated and understandable to human users. In the domain of communicative protocols, process calculus provides a tool for the high-level description of interactions, communications, and synchronizations between agents or processes. Agent modeling and communication languages make use of communicative acts to enable agents to communicate their intent for a specific service [2, 3]. For example, the Knowledge Query and Manipulation Language (KQML) and the Foundation for Intelligent Physical Agents agent communication language (FIPA-ACL) are two major developments in message exchange interaction protocols between agents [29, 56]. Such protocols aim to encode the communicative acts of an agent, as well as to model the agents'

knowledge using semantics. More specifically, the FIPA-ACL defines a set of (primitive and composite) communicative acts, along with a formal definition of the underlying semantic model. The semantic language formalism allows for the specification of the mental model of the agent (e.g., belief, uncertainty) and the effects of the acts on the interacting agents. A set of primitives defines how agents communicate information, including an *assertive inform* act, based on which an agent provides a message (in a proposition form), and a *directive request* act, which describes a request of sender (for an action from the receiver). Taking into consideration multi-agent coordination and social norms, Lightweight Coordination Calculus (LCC) can be used to specify the behaviours required of agents interacting in a given social context [61], as a form of 'electronic institution' [21]. This formalisation has been extended to model coordination and communication between multiple interacting actors in social context [53]. Our approach is inspired by such communicative protocols; however, our aim is to encode the interactions capturing the intent and the type of the exchanged information and not the effects of the act on the underlying semantic models of the interacting agents.

3 PROPOSED APPROACH: UNPACKING HUMAN-AI INTERACTIONS INTO INTERACTION PRIMITIVES

In this section, we present our approach towards a design space for HAI interactions. Our goal is to identify common interaction patterns from HAI interactions and specify them in terms of the type and intent of the communicated information. We provide a semi-formal definition of interaction primitives and we demonstrate how they can be used to define interactions between users and models. The outcome of the proposed approach is a list of interaction patterns and the actions that define them, along with a description of the interaction in terms of design and implementation aspects. We discuss how the proposed formalization can be used towards a design space.

3.1 Unpacking Human-AI interactions to patterns and primitives

Our approach aims to unpack HAI interactions into low-level communicative acts or *interaction primitives* which can specify the type and intent of the communication during a single interaction step. Such interaction steps are defined as user-model interactions, e.g., user provides an input, model provides an output, etc.. Our motivation is that unpacking HAI interactions can provide us with insights about the underlying design and technical challenges for HAI interaction patterns. In order to unpack HAI interactions into interaction primitives, we identify the interaction patterns between users and models and we specify the intent and type of the communicated information. The goal of this approach is to identify a set of communicative acts from different HAI interaction scenarios, towards a formalization for HAI interaction primitives. We demonstrate our approach using an interactive robot learning system for multimodal emotion recognition as a running example [83]. More use cases were selected for unpacking in order to cover different types of interactions, including XAI, HITL and hybrid intelligence interactions (See Appendix section A.1). We provide a description of the HAI interactions in the form of patterns and actions, and we highlight the design and implementation aspects of the interactions (Figure 2).

Description of the HAI interactions. The goal of this system is to collect and annotate human-generated data for an emotion classification model, through the human-robot interaction. In terms of interaction design, the interaction starts with an emotion elicitation session (user watches a clip which invokes a specific emotion). After the session, the user is asked to select their current emotional state from a list of emotions. This emotional state is provided as a model output (class) and it is used by the robot for the interaction as the ground truth emotion. The robot asks the user to walk towards and stand in front of it, demonstrating the selected emotion. The robot uses the gait/thermal data to predict the user's emotional state. If user's response (ground truth) is different from the predicted emotion, the robot asks the user about their current emotional state, annotating the collected gait/thermal data which are used to retrain the model.

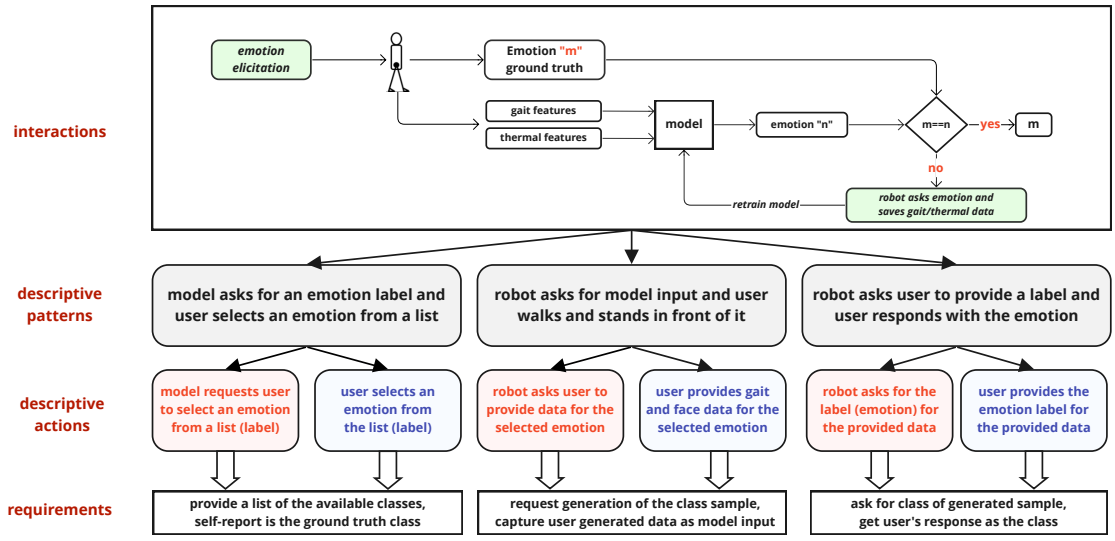

Fig. 2. Interactive robot learning [83] using descriptions of interaction patterns and primitives. The image shows the original interaction schema as a box and arrow diagram, followed by a high level description of the HAI interactions patterns, the unpacking into actions involving the exchange of information between the human and the system, and a description of the interaction requirements.

Interaction Patterns. We unpack the HAI interactions into three types of interactions between the user and the robot (model): *(a) class selection:* model asks user to select an emotion from a list (class) – user selects an emotion from the list, *(b) new class sample:* robot asks user to provide an input by demonstrating the selected emotion – user responds by walking and standing in front of the robot, and *(c) annotate sample:* model makes a prediction and asks the user for labeling, if prediction is different from ground truth – user provides the correct label by responding to the robot's question. During these interactions, user and robot exchange information in the form of *model input* (gait/thermal data) and *model output* (emotion from list, model prediction, robot question, user response). Model input is communicated as a set of raw data generated and captured during user's activity (walking and standing). Model output is communicated (a) through user's selection from a list, (b) implicitly through the robot's (model) prediction, and (c) as a user response to the robot's request during their interaction.

Design and implementation aspects. The robot runs an underlying emotion classification model which uses walking (gait data) and facial expressions (thermal data) as model inputs to predict the user's emotional state. The system utilizes the interaction with the user in order to dynamically improve the classification model by retraining on new (labeled) data. The design of the interactions enables the user to participate in the interactive learning process in an implicit way. The prediction model consists of two models (gait and thermal models) and the final prediction is estimated through a modified confusion matrix. After each interaction with the user, the model is retrained including the new data from the user as an input-output pair. The robot initiates the interactions to request for user's input (emotion label, input data and user response), and uses the responses to make interaction decisions; if model's prediction is different from user's input, the model implicitly requests the correct label from the user without communicating its

own prediction. A possible limitation of this implementation is that it highly depends on the quality of user's input. User may provide inaccurate information both during the selection (label) and the demonstration (sample) of the emotion.

3.2 Defining Interaction Primitives and Patterns for Human-AI interactions

Our motivation is to characterize HAI interactions in terms of the communicated information. In order to describe such interactions, we need to define how users and models interact with each other at a human-model level to either provide or request information. Based on the Semantic Interaction paradigm [74], both interactive agents should be able to communicate in an understandable and meaningful way. Following the taxonomy of instances for (hybrid) AI systems [70], model-specific information can be in different formats, including train/test data, learning and estimated parameters, symbols, rules, labels, and others. Moreover, user feedback and explanations can be communicated in a model-specific format. Based on the model specifications, user actions are translated through the user interface into machine-readable information. Considering these different types of communication (Figure 3), we define a set of *interaction primitives* to specify user/model actions in terms of the intent (provide or ask for information) and the type (format) of the communication. We demonstrate how the proposed formalization can be used to design interactions as exchanges of messages between the interacting agents. The descriptive formalization, along with a visual representation is shown in Figure 4. We provide a description of the definitions, as well as examples to demonstrate our approach.

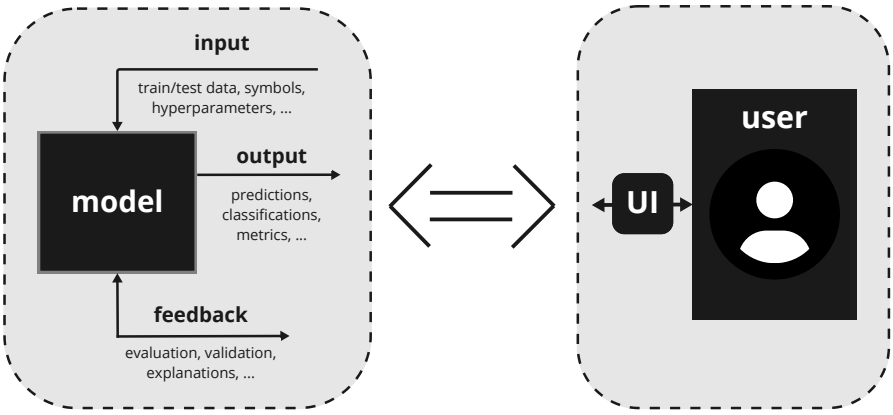

Fig. 3. Communication types during human-model interactions. AI models provide/receive information in a model-specific format: (a) input (test/train data, relations, hyperparameters, etc.), (c) output (labels, predictions, estimated parameters, projections, etc.) and (c) feedback (explanations, validation, requests, etc.). Users send/receive information through the UI which translates user actions to a model-understandable format and vice versa. Our approach aims to specify this communications using interaction primitives.

3.2.1 Interaction primitives and types. We define an *interaction primitive* as a low-level communicative act which specifies the type and intent of the communicated information. Considering the intent during an interaction step, the communicating agents can either *provide* or *request* information. We define a *provide primitive* as `P: provide(X:type+,Y:type*)` to describe the act of providing information (provide type `X` referencing type `Y`). Similarly, a *request primitive* is defined as `R: request(X:type+,Y:type*)` to describe the act of requesting information (request type `X` referencing type `Y`). For both primitives, the optional argument (`Y:type`) is used to reference an additional type. Considering the format of the information, human users and models can provide and

Definitions	
primitives	P: `provide(X:type+, Y:type*)`
	R: `request(X:type+, Y:type*)`
types	input: raw_data | fvector | state | query | mparams | ...
	output: class | regression | action | clusters | items | params | ...
	feedback: validate | evaluate | XAI.features | XAI.rules | ...
actions	`P | R ← op*`
	op: none | select(A,B) | map(A,B) | modify(A,B) | create(A) | ...
message	`<sender→receiver,action,[mod*]>`
	mod: <key: *var*, value= *any*>
pattern	message+

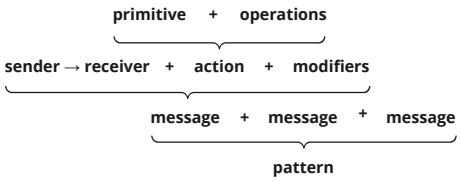

Fig. 4. Definitions and visual representation for interaction primitives, types and actions. An action is defined as a primitive specified by a set of operations. Actions are communicated through messages, by specifying the interacting agents and the modifiers of the message. Actions are reusable and can be used by different messages. Sequences of messages can form patterns of interaction.

request the following types of information: `input`, `output`, and `feedback`. The definition of the types (and subtypes) can inform the design and implementation of the specific interaction, since they characterize how users and models exchange information.

We provide an overview of the types (and their possible subtypes) with examples.

- `input` is information to be fed into a model. The *subtype* depends on the model type and architecture, including numeric vectors, world state information, images, video frames or sequences, user preferences, audio, text and so on. Model-specific parameters (hyperparameters) are also used as model input, e.g., learning rate, number of clusters, model sensitivity, etc.
- `output` is information coming out of a model — class labels, feature estimations, lists of recommendations, selected actions, generated images, projections of the input space, cluster labels, lower dimensional data and so on. This can also include learned parameters such as model weights, confidence values, loss etc.
- `feedback` refers to the additional information that can be provided both from users and models, including (requests for) evaluation, validation, explanations, etc. Explanations can be provided in different modalities, e.g., salient maps or natural language. Users can provide explanations to justify their decisions, as well as feedback for a model's decision. Similarly with model input/output, feedback is model-specific.

The proposed interaction primitives and types can describe different *actions* between users and AI models. Primitives capture the intent of the communication (provide/request) and types describe the format of the communicated information. For example, actions `u1: provide(X:input)` and `u2: request(Y:output, X:input)` can both describe a user communicating a model input type. However, the first action describes a user providing `X:input`, while the second one describes a user request for `Y:output` given the provided input. These actions are similar in terms of the provided input (`X:input`) but they differ on the type of the action (provide input vs. request output).

The selection of the types (and subtypes) depends on the model specifications and can inform design and implementation choices. For example, given that `provide(X:input.raw_data)` can describe a user who communicates a model input in the form of raw data, the interaction design (e.g., interface) should enable the user to interact with raw data (e.g., images). Primitives and types can also provide information about the requirements related to the goal of the interaction. Let us consider a face recognition model which takes as an input an image (raw data or extracted features) and detects faces (if any) in the form of bounding boxes. For this case, both of the defined actions `request(Y:output;X:input)` and `request(Y:output, [X:input, Z:output])` can describe a request

Examples of interaction primitives and types		
`provide(X:input)`	provide an input	upload/capture an image
`provide(X:output,Y:input)`	provide an output for a given input	detect a face in the image
`provide([X:input,Y:output])`	provide an input-output pair	show an image and the detected face
`request(X:output,Y:input)`	request an output for an input	ask if there is a face in a given image
`request([Y:input,X:output])`	request an input-output pair	ask for an image with a detected face (if any)
`provide(Z:output,[Y:input,X:output])`	provide output for an input-output pair	modify an existing bounding box on an image
`request(F:feedback,Y:output)`	request feedback for the given output	ask for confirmation about a bounding box
`request(X:[input])`	request a set of inputs	ask for a set of input images

Table 1. Examples and descriptions of interaction primitives and types

for an output (detected face image). The first one is an output request for a given input image, and can describe a face detection interaction. The second action is an output request given an an input-output pair (image - bounding box) and can describe a modification request for the model's detection. Table 1 provides examples of interaction primitives and types along with a description in the context of face detection.

3.2.2 Actions, operations and modifiers. In order to specify an interaction between a user and a model for a given interaction context, we define a message as `msg: <sender→receiver,action,[mod*]>` to describe the communication of an action from a sender to a receiver. An action is defined as `P | R ← operations` and specifies a primitive action by adding a description of how the arguments need to be communicated, through a set of `operations`. The operations contextualize an action by specifying the preconditions for its communication in a given interaction context. More specifically, the operations define how the argument is being created and describe the relations between multiple arguments. A list of operations includes, but is not limited to:

- `select(A,B)`: argument `A` is selected (from a given set `B` - optional argument) – this operation can describe the selection of an item from a list or set of choices, i.e., recommendations, labels, samples, etc.
- `map(A,B)`: argument `A` is mapped to argument `B` – this operation can be used to describe a model prediction (e.g., classification, regression, clustering), human labeling, evaluation, etc.
- `modify(A,B)`: modify argument `A` to argument `B` – this operation can describe a modification of a sample (modify input image), an alternate decision (change label), etc.
- `create(A)`: create new argument `A` – this operation can describe data acquisition or generation (e.g., image, sound), a human annotation (e.g., new label), etc.

Finally, a set of `modifiers` (mod: <key: *type*, val=*any*>) can be used to further characterize the message in terms of interaction requirements, as a free-form annotation feature. Modifiers can provide information about the interaction modality, interface elements (e.g., buttons, forms, etc), type of communication (e.g., explicit vs. implicit), etc. Based on the above, the definition of an action includes the primitive (provide/request), the type (input, output, feedback), the operations needed to communicate the types, as well as additional information for the interaction through the modifiers, and describes the communication of an action as a single message from a sender to a receiver. A sequence of actions for a given interaction context is defined as an *interaction pattern*.

The proposed formalization allows for a custom definition of an action and its specification as a message for a given interaction context. For example, we can define the action **`req-new_sample(M)`** ≡`request(M:input.raw_data)`←`create(M)` to describe a request for a generated input in the form of raw data. Sequences or exchanges of messages can be used to define interactions between users and models for a given context. For interactions with a face recognition model, we can define a message `msg: <model→user,` **`req-new_sample(M)`**`,[M:image]>`

to describe the model's request for an image from the user. As a response to this request message, the user could respond with either:

- msg1: <user→model, **generate-sample(M)**, [M:image]>,
  where: generate-sample(M)≡provide(M:input.raw_data)←create(M), or
- msg2: <user→model, **req-gsample_class(M,L)**, [M:image]>,
  where: req-gsample_class(M,L) ≡ request(L:output.label,M:input.raw_data)
  ←[create(M),map(M,L)]

Based on the first message, the user responds by providing the requested input (create/capture image), while with the second message, a request is made to the model for face detection given the generated input (map/assign the captured image to a label). The same action can be communicated by different messages. A message specifies the communication of an action from a sender to a receiver in a given interaction context. For example, a speech recognition model can communicate its request for user-generated input (speech) using msg': <model→user, **req-new_sample(M)**, [M:speech]>. From this, a set of actions can be defined as a vocabulary which can be used in different HAI interactions. Such actions and messages can be used to design interaction patterns as sequences of messages between users and models. Given the message definitions above, we can define query_input≡[msg,msg1] as a query pattern to describe the request and communication of a model input, while query-label_input≡[msg,msg2] could describe an interaction where the user awaits for the model's decision based on a new sample. These patterns serve different interaction goals and also require different implementation. In the next section, we provide a set of action definitions and interaction patterns from existing HAI interactions and frameworks.

3.3 Designing interaction patterns using actions and messages

In this section, we describe how the proposed primitives can be used to define actions and interaction patterns. Such patterns serve a given goal during the interaction and can be applied for the design of other interactions. For the *interactive robot learning for emotion recognition* (Section 3.1), we define the following messages and actions (Table 2):

	Message	Action definition
A1	<model→user,**req-class_selection(Y,L)**, [Y:reqSelfReport;L:listEmotions]>	req-class_selection(Y,L) ≡ request(Y:output.label, L:[output.label]) ← select(Y,L)
A2	<user→model,**select-class(Y,L)**, [Y:SelfReport;L:listEmotions]>	select-class(Y,L) ≡ provide(Y:output.label,L:[output.label]) ← select(Y,L)
A3	<model→user,**req-new_class_sample(X,Y)**, [X:reqWalkStand,Y:SelfReport]>	req-new_class_sample(X,Y) ≡ request(X:input.raw_data,Y:output.label) ←create(X), map(X,Y)
A4	<user→model,**generate-class_sample(X,Y)**, [X:WalkStand,Y:SelfReport]>	generate-class_sample(X,Y) ≡ provide(X:input.raw_data,Y:output.label) ←create(X), map(X,Y)
A5	<model→user,**req-sample_class(X,Y)**, [X:WalkStand;Y:reqSelfReport]>	req-sample_class(X,Y) ≡ request(Y:output.label,X:input.raw_data) ← map(X,Y)
A6	<model→user,**annotate-sample(X,Y)**, [X:WalkStand;Y:SelfReport]>	annotate-sample(X,Y)≡ provide(Y:output.label, X:input.raw_data) ←map(X,Y)

Table 2. Messages and action definitions for the interactive robot learning interactions

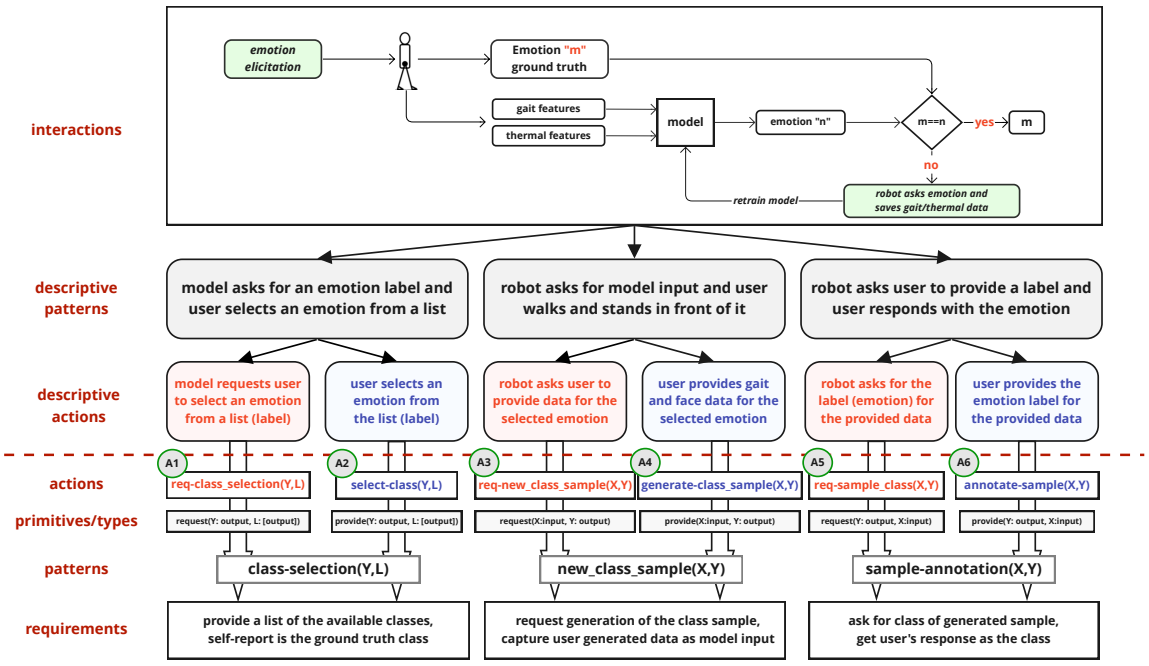

Fig. 5. Description of the HAI interactions using definitions of interaction primitives and patterns. The diagram illustrates our unpacking approach and the process of describing interaction primitives and patterns in terms of the interaction requirements.

Interaction Patterns. The defined messages and actions can semi-formally specify: (a) the intent of the sender's act (primitive), (b) the type of the communicated information (types and operations), and (c) how the information is communicated (modifiers). Based on the unpacking of the interactions, we define the following interaction patterns: `class_selection≡ [A1,A2]` , `new_class_sample ≡ [A3,A4]` and `sample_annotation ≡ [A5,A6]` (Table 3). All patterns describe a query-response interaction initiated by the robot (model). The goal of the first pattern is to set the target class, without requiring any information about the model input. The class represents the user's emotional state selected from a predefined list of emotions (after an emotion elicitation activity). The goal of the second pattern is to receive a model input for the target class (emotion), resulting to a training example (input-output pair). The robot asks the user to demonstrate the selected emotion (model input - waling and standing). For the third pattern, the model captures the gait/thermal data and makes a prediction for the user's emotion. This prediction is not communicated to the user but it is used for the design of the interaction; if the prediction is inaccurate, the robot interacts with the user and uses the response to retrain its model. These patterns can be used to design the interactions between the human user and the model (Figure 5).

Design and implementation aspects. The selection of the patterns and the actions that define them can provide insights about the design and implementation aspects. For example, designing the `class-selection` interaction requires an interface for the user to choose an emotion so it can be communicated to the model (robot). For the `new_class_sample` pattern, the model needs to capture the gait/thermal data and process them for a model prediction (feature extraction). In terms of technical aspects, the implementation does not consider user's input uncertainty, making the assumption that the user provides the correct data (model input and output). Such choices can

affect both the performance of the model and its interaction with the user. Different actions or patterns can lead to different types of interactions between users and models, e.g., negotiation, which may require different methods for implementation, e.g., shared decision making.

pattern	actions	message description
class-selection	`req-class_selection(Y,L)`	model asks user for a class from list
	`select-class(Y,L)`	user selects a class from a list
new_class_sample	`req-new_class_sample(X,Y)`	model asks user to provide a sample for the given class
	`generate-class_sample(X,Y)`	user provides a sample for the given class
sample-annotation	`req-sample_class(X,Y)`	model asks user to annotate sample
	`annotate-sample(S,M)`	user provides a label for the sample

Table 3. Interaction patterns for the interactive robot learning interactions

Based on the unpacking process of HAI interactions to primitives and patterns on this running example, we compiled a list of three patterns, and the actions that define them, which can describe the interactions between the user and the robot. Following the proposed formalization for actions and patterns, this approach aims to (a) characterize existing interactions, and (b) modify or design new interactions. Our motivation is to compile a list of commonly used actions and patterns, towards a design space for HAI interactions.

3.4 A list of interaction patterns and actions

Based on the unpacking of HAI interaction use cases (Appendix A.1) and review on frameworks (Section 2), we compiled a list of actions, messages and interaction patterns. We provide a set of action definitions with their description (Table 4). Such actions can be further specified as messages and used for a given interaction context. For example, `annotate-sample(X,Y)` describes the annotation of a sample during a classification task, as the mapping of input sample X to output label Y. A similar action, `show-policy(S,A)`, describes an RL policy as the mapping of input state S to output action A. This list is not exhaustive since it does not cover all possible actions and patterns, but rather provides examples from a range of HAI interaction scenarios. More actions can be defined following the proposed formalization, resulting to an extendable library of actions. While there are different ways to define these actions and messages for the selected use cases, the goal of this approach is to specify the intent and type of the communicated information considering existing interaction concepts and HAI interaction paradigms.

Following the proposed definitions, we provide a collection of interaction patterns (Table 5). Each pattern is described as a sequence of messages based on the action definitions and the unpacking process of the selected HAI interactions. The selection of the messages and actions play an important role to the definition of an interaction pattern. For example, `req-class_selection` and `req-sample_class` can both describe a model's request for an output. However, the first action specifies the selection of a class from a list, while the second action requires a class given an input sample. The selection of an action for a given pattern depends on the goal of the interaction, as well as the design and technical requirements. Combining different actions can lead to patterns with different interaction goals. For example, model queries for (informative/evaluative) advice aim to improve model's performance, while XAI-based interaction patterns can be used to support the user by providing justifications about model predictions. This collection captures a range of interaction patterns in terms of the interaction concept and requirements. Combinations of interaction patterns can serve multiple goals and concepts. For example, providing explanations to users (XAI) can enhance the quality of human feedback (HITL) resulting to collaborative learning systems. The long-term goal of this research is

Action Definition	Description
req-class_selection(Y,L) ≡ request(Y:output.label,L[output.label]) → select(Y,L)	request the selection of a class Y from list L
select-class(Y,L) ≡ provide(Y:output.label, L:[output.label]) → select(Y,L)	select a class Y from a list L
req-new_class_sample(X,Y) ≡ request(X:input.raw_data|fvector, Y:output.label) → create(X),map(X,Y)	ask for a new sample X for a given class Y
req-class_sample(X,Y) ≡ provide(X:input.raw_data|fvector, Y:output.label) → select(X), map(X,Y)	select a sample X of a given class Y
req-sample_class(X,Y) ≡ request(Y:output.label, X:input.raw_data|fvector) → map(X,Y)	ask for the class Y of a given sample X
req-gsample_class(X,Y) ≡ request(Y:output.label, X:input.raw_data|fvector) → create(X),map(X,Y)	request the annotation of a generated sample
req-sel_sample_class(X,Y) ≡ request(Y:output.label, X:input.raw_data|fvector) → select(X),map(X,Y)	request the annotation of a selected sample
annotate-sample(X,Y) ≡ provide(Y:output.label, X:input.raw_data|fvector) → map(X,Y)	annotate a given sample
show-policy(S,A) ≡ provide(Y:output.action, X:input.state) → map(S,A)	show selected action A for state S
give-evaluative_advice(S,A,R) ≡ provide(R:feedback.eval, [X:input.state,Y:output.action]) → select(R), map(S,A,R)	select a reward R for the mapping from S to A
modify-prediction(X,Y,Z) ≡ provide(Z:output.label,[X:input.raw_data|fvector, Y:label]) → modify(Y,Z), map(X,Z)	modify prediction of X from Y to Z
show-candidate_samples(CS,S) ≡ provide(CS:[input.raw_data], S:[input.raw_data]]) → select(CS,S)	show a list of candidate samples CS based on sample S
select-sample(X,CS) ≡ provide(X: input.raw_data, CS:[input.raw_data]]) → select(X,CS)	select sample X from a list of samples CS
modify-sample(X,M) ≡ provide(M:input.raw_data,X:input.raw_data) → modify(X,M)	modify a sample from X to M
generate-sample(X) ≡ provide(X:input.raw_data) → create(X)	modify a sample from X to M
modify-mparams(P,M) ≡ provide(M:input.mparams,X:input.mparams) → modify(X,M)	modify model parameter from P to M
modify-features(X,M) ≡ provide(M:input.fvector,X:input.fvector) → modify(X,M)	modify a feature vector from X to M
req-prediction_evaluation(X,Y,F) ≡ request(F:feedback.eval,[X:input.raw_data, Y:output.label]) → select(F), map(X,Y,F)	ask for feedback F to evaluate the mapping of X to Y
evaluate-prediction(X,Y,F) ≡ provide(F:feedback.eval,[X:input.raw_data, Y:output.label]) → select(F), map(X,Y,F)	select feedback F to evaluate the mapping of X to Y
show-prediction_XAI(X,Y,F) ≡ provide(F:feedback.XAI,[X:input.raw_data, Y:output.label]) → map(F,X,Y)	provide explanation for the mapping of X to Y

Table 4. A list of action definitions and their description, extracted by unpacking existing HAI interactions into interaction primitives.

the formalization of a new design space for HAI interactions which will support designers to explore, modify, and apply interaction patterns by providing design and implementation choices towards the prototyping of new types of interactions between human users and AI models.

Patterns	Actions	Description
class-selection	req-class_selection(Y,L)	request for a class from a list
	select-class(Y,L)	select a class from a list
new_sample	req-new_sample(X)	ask for a new input sample
	generate-sample(X)	generate an input sample
new_class_sample	req-new_class_sample(X,Y)	request a new sample for a given class
	generate-class_sample(X,Y)	generate a sample for a given class
sample-annotation	req-sample_class(X,Y)	ask for the class of a given sample
	annotate-sample(X,Y)	select a class for a given sample
new_sample-annotation	req-new_sample(X)	ask for a new input sample
	req-gsample-class(X,Y)	ask for the class of a new sample
candidate_samples	req-candidate_samples(CS,S)	ask for a set of candidate input samples
	show-candidate_samples(CS,S)	show a set of candidate input samples
sample-modification	req-modified_sample(X,M)	ask for a modified input sample
	modify-sample(X,M)	modify an input sample
feature-modification	req-modified_feature(X,M)	ask for a modified feature vector
	modify-feature(X,M)	provide a modified feature vector
parameter-modification	req-mparam-modification(P,M)	ask for a modified model parameter
	modify-mparam(X,M)	modify a model input parameter
prediction-modification	annotate-sample(X,Y)	provide a label for a sample
	modify-prediction(X,Y,M)	modify a prediction of a given sample
policy-visualization	show-policy(S,A)	show selected action for current state
informative_advice	req-informative_advice(S,A,B)	ask for informative advice based on state-action
	give-informative_advice(S,A,B)	modify (or not) the selected action
evaluative_advice	req-evaluative_advice(S,A,B)	ask for evaluative feedback for state-action
	give-evaluative_advice(S,A,B)	evaluate the state-action pair
prediction-based_XAI	req-prediction_XAI(X,Y,F)	ask for explanations for a given input-output
	show-prediction_XAI(X,Y,F)	show explanation for a given input-output pair
outcome-evaluation	req-outcome_evaluation(Y,F)	request evaluative feedback for a given outcome
	evaluate-outcome(Y,F)	provide evaluative feedback for a given outcome
prediction_parameters	req-prediction_params(X,Y,P)	request predictions with model output parameters
	show-prediction_params(X,Y,P)	show predictions with model output parameters
turn_taking-evaluation	generate-and-turn(X,Y)	request a modified sample based on a generated input
	capture-and-generate(X,Y)	provide a modified sample based on input
	evaluate-outcome(Y)	provide evaluative feedback based on outcome
prediction-with-XAI	select-sample(X)	select a sample
	show-prediction_XAI(F,X,Y)	show explanation for the sample prediction
	modify-annotation(X,Y,M)	modify the outcome based on the input-output pair
recommendations	req-recommendations(M,R)	modify a model input parameter
	show-recommendations(M,R)	show recommended items for a given model input
	evaluate-recommendation(S,V)	evaluate a selected recommended item (accept/reject)

Table 5. A list of interaction patterns as sequences of the defined actions/messages.

4 FROM INTERACTION PRIMITIVES TO A DESIGN SPACE FOR HAI INTERACTIONS

In this section, we discuss how the proposed formalization can be used towards the definition of a design space for HAI interactions. We demonstrate how the proposed interaction primitives and patterns can be used as design materials to prototype HAI interactions. We highlight how differences in patterns can serve different interaction goals and concepts.

The goal of the proposed design space for HAI interactions is to support AI designers and practitioners by providing appropriate design and implementation choices for a given interaction concept. Based on the literature review and the unpacking of existing HAI interactions, we provide an overview of how interaction patterns can be designed for given interaction paradigms, aiming to bridge the gap between high-level guidelines and implementation requirements.

4.1 Interaction primitives and actions as design materials

Human-centered AI approaches, including explainability, transparency, interactivity, and human control, have provided opportunities to design new types of interactions, e.g., model auditing and contestation, negotiation, shared decision making, and others. HAI interactions can utilize the available information provided by the models, apart from decisions and predictions. For example, model uncertainty can be measured, communicated and used as a design material through transparency [11]. AI designers and practitioners need to consider interactions with AI as a design material, based on its capabilities, limitations, and the challenges that may arise while designing for transparency, unpredictability, learning, and shared control [37]. Our proposed formalization for interaction primitives and patterns aims to enable designers and AI practitioners to design HAI interactions by exploring appropriate patterns for a given set of design and technical requirements. The following example illustrates two alternatives of a query pattern, where the user queries the model for a prediction using a model input - `req-sample_class(X,Y)`, receives the model's prediction, and provides a modified prediction - `modify-prediction(X,Y,Z)`. Using the same user actions, we can define two query patterns based on two different model actions (Figure 6):

- **`annotate-sample(X,Y)`** ≡ `provide(Y:output,X:input)`⟵`...`, and
- **`req-modified_prediction(X,Y,Z)`** ≡ `request(Z:output,[X:input,Y:output])`⟵`...`

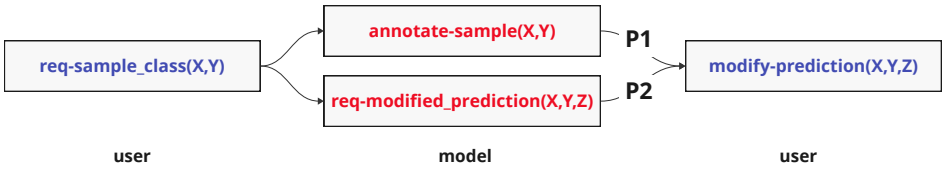

Fig. 6. Query Patterns. The user asks for a model prediction for a given input. Based on pattern P1, the model provides the prediction and the user chooses to modify the prediction. Based on pattern P2, the model makes an explicit request for a modified prediction.

In the first case (P1), the model provides the user with a prediction on user's input and the user contests the prediction by providing an alternate output. The second case (P2) describes a model which queries the user to provide an alternative output given its prediction on the user's input. In terms of design and implementation choices, the first query alternative may require a mechanism to decide who makes the decision based on the quality of user/model predictions and how to update the model based on user's prediction (hybrid decision making). For the second query, the model is designed to explicitly ask the user for an alternative output and could be an example of active learning, where the model requests human annotation for (uncertain) predictions. In that case, human's decision should have a larger effect on the model updates, compared to the first version. Through these two versions of a query, we demonstrate how primitives can be used as design materials to generate different patterns given an interaction goal. Both patterns could describe the interactive control feature for the design of *contestable AI* interactions [6], which enable the users to intervene and modify a decision made by the system.

4.2 Prototyping interactions

A key aspect of the proposed design space is the ability to create and explore alternatives of interactions by selecting different patterns or actions. We consider a set of alternative designs for the interactive robot learning scenario [83], using the defined interaction primitives and patterns as design materials. We demonstrate how different sequences of patterns can result to different types of interactions. For each design alternative, we provide a description of the design and technical aspects, as well as the interaction concept (Figure 7). The first design (D1) represents the original interaction

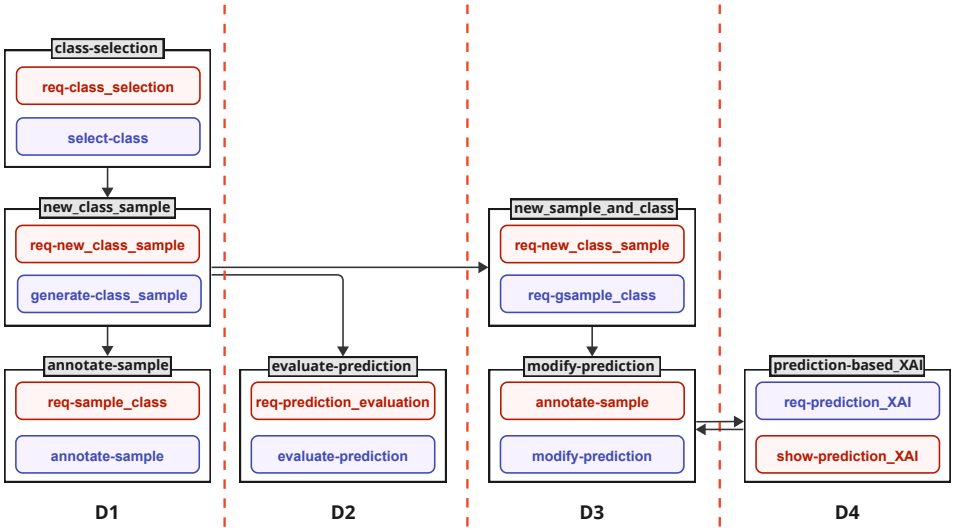

Fig. 7. Design alternatives for the interactive robot learning interactions [83]. Different sequences of patterns result to a set of design alternatives for different interaction concepts. Each alternative is related to specific design/implementation aspects and requirements. D1 describes the original design. D2 describes a robot-initiated interaction for user feedback. Based on D3, user asks the robot for its prediction based on the generated sample which can be modified. D4 adds an XAI-based interaction for model explanations for the robot's prediction.

design (Figure 2); a robot-initiated interaction where the user provides an annotated sample in a query-response manner. The second design (D2) describes an interaction where the robot communicates its prediction for the user's emotion, followed by a request for user's validation. This is achieved by replacing the last pattern `(annotate-sample)` with a pattern for evaluating the model's prediction `(evaluate-prediction)`. The third design (D3) can describe a system where the user asks the robot to make a prediction which they can modify. This design introduces two patterns to allow the user to request model's prediction for the generated sample `(new_sample_and_class)` and modify it `(modify-prediction)`. The fourth design (D4) includes an additional XAI-based interaction pattern, where the model provides explanations about its prediction to the user `(prediction-based_XAI)`.

These alternatives can characterize different types of interactions with respect to the goal of the interaction. The original design is proposed as an interactive robot learning approach where the goal is to evaluate and improve the model's predictions through its robot-initiated interactions with the user. The second design requires the user to be more active in the interaction through an interaction pattern for model's prediction and user's feedback (evaluation). Based on the third design, the user initiates the interaction for the model's prediction. Finally, adding the XAI interaction pattern enables the user to further interact with the robot for explanations. These designs are also related to specific

design and implementation aspects. The original design assumes that user's provided sample and self-reported emotion are accurate. The second one requires an interaction where the robot communicates the prediction and asks for user's evaluation. Such an interaction requires an appropriate learning mechanism which can integrate user's feedback to the learning process. The XAI-based interaction requires the implementation of a specific explanation method. These alternatives can also be related to different roles of users involved in the interaction. The first design (D1) has been proposed for an end-user who implicitly participates in the model training process. The fourth alternative (D4) could be a design for an interaction between the model and the developer for model evaluation, where XAI is used to enhance the user's understanding about the system capabilities, e.g., identify "hard-to-predict" emotions.

Considering multi-user HAI interactions, different types of users can participate in the interaction for different goals. For example, the robot-based game interactions [69] are divided into two interaction loops; player-AI and supervisor-AI interaction. Each loop serves a specific interaction goal. The player-AI interaction goal is to elicit feedback from the player implicitly in order to personalize the model's decisions, while the supervisor-AI loop aims to enhance safety through the interventions of a supervisor using a transparent interface. Considering these, different interaction patterns should be used to serve each goal (Figure 8).

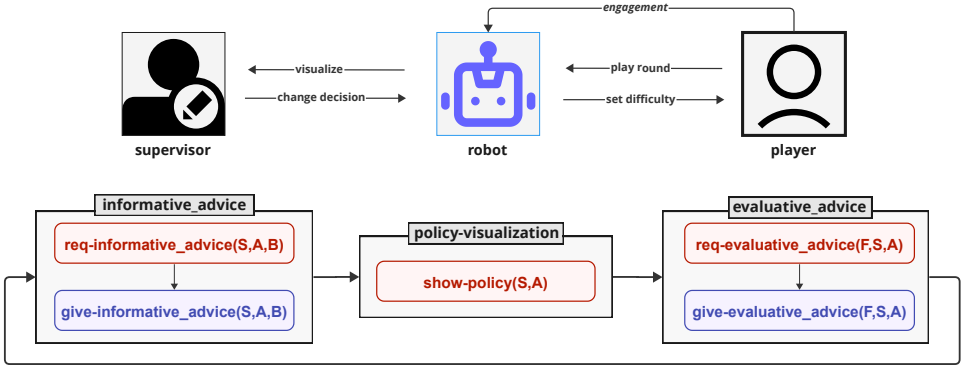

Fig. 8. Interaction Patterns for the robot-based multi-user interaction [69]. The robot visualizes its policy to the supervisor (`policy-visualization`) who can alter it (`(informative_advice)`). The robot receives implicit feedback from user's engagement `(evaluative_advice)`. Both types of feedback are integrated to model learning updates.

The patterns describe the interactions between the robot, the player and the supervisor. The robot adjusts its policy (difficulty selection) based on user's performance and engagement, and the supervisor's interventions. Two main design aspects of the proposed system are (a) transparency and explainability and (b) user feedback and control. Considering the player-AI interactions, the model communicates the selected difficulty through the robot's announcement and the player provides implicit feedback through task performance and engagement. For the supervisor-AI interaction, model's transparency through the UI aims to enhance user's decision making by providing appropriate interventions. Considering both types of interactions and their patterns, a learning mechanism is required to learn from both types of users in an online way so the robot can dynamically improve its policy and select the appropriate levels of difficulty. The goal of the interaction is to include both types of users in the personalization process.

Designing multi-user interactions may also require combining interaction patterns for different goals. Considering the framework for contestable AI [6], we provide a description of possible interactions during model contestation,

highlighting the different interaction concepts between users and the AI model. The framework follows the paradigm of mixed-initiative interaction, based on which human controllers and decision subjects can both participate in the decision making process. Decision subjects can interact with the system to negotiate a decision which affects them. Such decisions may be the outcome of decision support interactions between the model and the human controller (semi-automated decisions). In order to support such types of interactions, the proposed framework provides the following features: *interactive controls*, *explanations*, and *intervention requests*. Interactive controls enable both types of users to provide feedback to the system for different purposes. Explanations are used to provide justification for the model's decisions and aim to support the semi-automated decision making process. Intervention requests enable the decision subject to initiate a model auditing process. XAI-based interaction patterns can be used to: (a) make a user aware of a decision made by the system, (b) inform the user about how to contest a model decision and (c) provide an explanation to justify the decision of the system. HITL-based interaction patterns can provide both types of users with the ability to negotiate and even override AI decisions (interactive controls). Moreover, collaborative learning interactions can enable both users and system to learn from their interactions and augment their decision making towards hybrid intelligence.

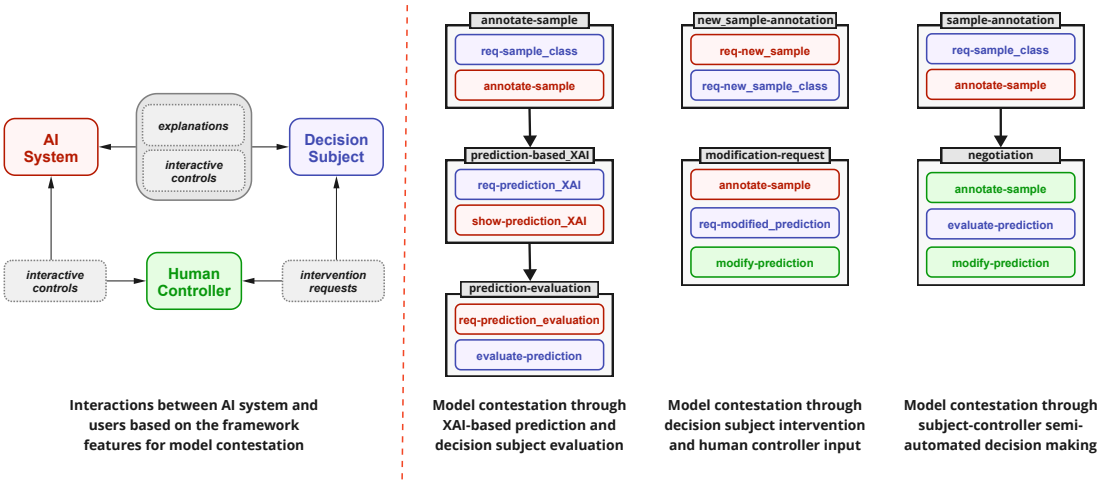

Fig. 9. Examples of Interaction Patterns for contestable AI interactions [6]. We describe three examples of interactions based on the proposed framework for mixed-initiative interactions between the AI system, the decision subject and the human controller. Each interaction describes a different contestation aspect using interactive controls, explanations and intervention requests.

Considering the above, we provide examples of interaction based on our defined actions and patterns (Figure 9). The first example is a combination of three interaction patterns and describes the communication between AI and the decision subject. During the interaction, the user asks for the model's prediction for a sample (`sample-annotation`) and for explanations for this prediction (`prediction-based_XAI`). The subject can utilize the explanations to evaluate the outcome. if needed (`prediction-evaluation`). The second example is a multi-user interaction, where the decision subject needs to generate a new sample (e.g., submit a form) and ask for a decision (`new_sample-annotation`). The AI provides its decision to subject who makes a request to the human controller for the modification of the decision (`modification-request`). Finally, the third example describes a semi-automated decision making process, where

the human controller can provide a decision (considering the AI's output) and modify it based on the decision subject's evaluative feedback for the model's prediction (`negotiation`).

4.3 Interaction Patterns and Requirements

We demonstrated how the proposed formalization can be used towards prototyping HAI interactions using patterns. In order to implement such interactions, we need to consider the interaction goals that each pattern can serve. Towards this, based on our literature review (Section 2) and the extracted patterns and actions (See Tables 4 and 5) from the unpacking process of HAI interaction use cases (See Appendix A.1), we provide an overview of how interaction patterns can be used in the context of different concepts for XAI-based, HITL-based, and HI-based interactions. The goal of the overview is to identify possible relations between interaction patterns and requirements.

4.3.1 XAI-based interactions. Human-XAI interactions can be designed for several interaction concepts and goals (e.g., debugging, persuasion, decision support, etc.). Designing explainable and transparent models is not trivial, especially while considering the various parameters that can affect the interaction, e.g., user's expertise, perception and understanding, cognitive load, preferences, etc. From the unpacking process, we identified the following XAI-based patterns and interaction goals:

- *XAI-based interactions can manage user's expectations about the AI model's behavior.* A Meeting Scheduling Assistant [40] (See Appendix A.1 - Figure 16 for description) uses explanations to calibrate user's trust and expectations about its model predictions. The model predicts if an email is a meeting request to help the user make a decision. The `prediction-with-XAI` pattern enables the model to be transparent by communicating its accuracy rate along with the prediction. The system visualizes the model's accuracy rate, as well as a set of prediction/explanation examples with different levels of uncertainty to enhance user's understanding.
- *XAI-based interactions can enhance user's perception about the model's performance.* In the context of active learning for emotion recognition [35] (See Appendix A.1 - Figure 13 for description), XAI-based interactions aim to facilitate the selection of appropriate samples for model refinement. Based on the unpacking process of the proposed system, we identify two patterns with different goals: the `prediction-parameters` pattern aims to support the selection of appropriate samples for annotation by visualizing the confidence of predictions, and the `prediction-based_XAI` pattern is used to support the annotation process by providing additional information about the model's prediction for a given sample through visual explanations.
- *Model transparency can support human trainers while providing feedback to iML models.* Policy visualization [12] (See Appendix A.1 - Figure 12 for description) serves as a transparency method to engage the user to provide feedback to the model during task performance. The model utilizes user feedback to facilitate its learning process, i.e., faster convergence to the optimal policy. Based on our unpacking approach, the `policy-visualization` pattern is used to communicate the model's current policy, by visualizing the current state (input) and the selected action (output).
- *Explanations can justify model's prediction based on user preferences.* For an explainable Music Recommendation system [47] (see Appendix A.1 - Figure 15 for description), explainable user (preference) models are used to enhance user's perception about their own preferences and how these affect model's recommendations. Considering this, the `prediction-based_XAI` pattern is used to provide predictions and justify them through explanations based on user's preferences. In terms of design aspects, different visualizations (and explanation methods) are required considering the individual characteristics of users, e.g., need for cognition.

4.3.2 HITL-based interactions. The goal of HITL methods is to efficiently integrate the human user to the learning and decision making process of an AI system. Human users can participate in the model's development and deployment phases. The selection of appropriate iML/HITL methods and approaches depends on several aspects of the interaction, including user role and expertise. We present a set of HITL-based patterns extracted from the unpacking process, considering the different interaction goals.

- *HITL methods can be used for interactive data collection and labeling.* For the interactive robot learning scenario for emotion recognition (Figure 2), the model utilizes human-robot interaction data in order to evaluate its predictions and collect training data. Model re-trains without making the user aware of their participation in the data collection, annotation and training processes. Based on our unpacking, `sample-annotation` defines a human labeling action given a generated sample.
- *HITL methods can be used for model improvement by relabeling uncertain predictions.* In the context of active learning, the NOVA system (See Appendix A.1 - Figure 13 for description) asks the user to select uncertain samples and modify their predictions. In terms of the interaction design, the `prediction-modification` pattern enables the user to validate the model's prediction or provide a new label. Manual corrections are used to update the model. Since the user makes the final decision (label), XAI methods are used to enhance user's perception and, thus, the quality of feedback.
- *Feedback interfaces should be user-friendly and intuitive in order to ensure human feedback quality.* Training an RL agent through human advice (Appendix A.1 - Figure 12) requires an appropriate learning methods to integrate human feedback. The interaction supports two types of feedback. The `informative-advice` pattern is used to receive human advice in the form of a corrective action and the `evaluative-advice` pattern describes the evaluation of the model's decision in the form of binary feedback. These types of feedback are integrated through different feedback interfaces and learning methods (policy/reward shaping).
- *Users can control model predictions and parameters.* For the Meeting Scheduling Assistant (Appendix A.1 - Figure 16), HITL methods enable the user to (a) provide feedback for a prediction by accepting or rejecting it, and (b) control the model's sensitivity parameters through a UI slider, affecting the model's predictions. The `modify-prediction` action enables the user to validate (or not) a prediction and the `modify-mparams` action allows the user to adjust the model's sensitivity parameters until they are satisfied with the model predictions.

4.3.3 Collaborative Learning and Hybrid Intelligence. The goal of collaborative learning and hybrid intelligence interactions is to enable both humans and machines (AI systems) to learn from each other in a collaborative manner. Collaborative learning interactions can be designed by combining XAI-based and HITL-based interactions, enabling both users and AI models to exchange information while solving a task. Based on our unpacking process, we identify and discuss interaction patterns used in collaborative learning and hybrid intelligence interactions.

- *User can control and evaluate the collaboration with a model.* In the context of a robot-based collaborative sketching interaction [43] (See Appendix A.1 - Figure 14 for description), both user and robot work together during the co-ideation process in a turn-taking interaction. Both agents generate ideas building on their partner's generated sketch. The model uses the captured image to generate a variation of the user's sketch - to provide alternative ideas. During the interaction, the user can control what the robot will capture as an input by moving the robot and also provide feedback for the generated outcome. The `turn_taking-evaluation` pattern is repeated until the user is satisfied. The model uses image classification for the user's input and generates a variation of

this. The goal of such interactions is for the user to explore and identify new insights, rather than to identify a correct solution.

- *Model can support the user through semi-automated decision making.* An interactive sound segmentation system [39] (Appendix A.1 - Figure 11) enables the user and the model to work together in order to complete an audio segmentation and annotation task. The model supports the user by providing a list of candidate samples which can be selected, edited and annotated by the user, towards a collaborative interaction. The `candidate-samples` pattern requires a model mechanism to identify the candidate samples and a proper visualization to highlight the segments. Based on this visualization, users provide feedback to adjust the model's predictions.
- *Explanations can be provided both by users and models to justify their predictions.* In a game-based scenario [34] (See Appendix A.1 - Figure 17 for description), XAI and iML methods are used to enhance user's trust about the model's decisions and allow for corrections. The `turn-taking_XAI` pattern describes an interaction where both user and model communicate their prediction justification in the form of rule-based XAI. In terms of the design and implementation of this interaction, an appropriate visualization of the rules is needed to enable the user to understand the reasoning of the model in order to provide appropriate modifications and justification.

Based on this overview, we can observe that each pattern can serve a specific goal within the interaction. For example, considering the XAI-based patterns, the `prediction-with-XAI` pattern is used to calibrate end-user's trust, while `prediction_XAI` is used to enhance user's understanding about a prediction. For HITL-based patterns, `annotate-sample` enables a user to improve a model by providing annotations, while `modify-mparams` is used as a control pattern which enables the user to alter the model's predictions until the user is satisfied with the decision. Considering possible commonalities and differences between patterns and goals, we envision a design space which can provide suggestions for interaction patterns based on a given interaction concept. This would allow for fast prototyping of interactions using patterns, considering the interaction goals and requirements.

5 DISCUSSION

5.1 Limitations

A basic limitation of our proposed formalization is that it describes interactions without providing information about how the interacting agents are affected by the communication of a message, or their underlying processes (e.g., predict or fit/update) and the dataflow during the interaction. Moreover, the current formalization allows for multiple ways to define a given interaction. In order to support the design and prototyping of HAI interactions following our approach, we need to introduce a formalization of the primitives and actions as design materials, including instances of objects and agents (user profiles, model cards, data sheets) and their operations (model operations, human decisions, etc.). For example, the `new_sample-annotation` pattern (Table 5) includes an action for the generation of the new sample and an action for the sample annotation. These actions are related to specific model operations; generating a sample needs a `preprocessing` step so it can be used by the model for a `model.predict` operation. Moreover, retraining a model by altering its predictions, e.g., `modify-prediction`, can be implemented as a `model.fit` operation using updated training data. Such formalization will allow for the design of interactions between multiple agents and objects (users with different roles, models/data with different levels of access, etc.). Another limitation of the current approach is that the XAI-based interactions can be further described considering the explanation method. Our current approach considers XAI as a separate type for the defined primitives. For example, LIME-based visualization [35] (Figure 13) communicates the important features/pixels of an image which affect the prediction. With our current

approach, this interaction is described using the `feedback.XAI` type and the LIME method is defined as a modifier. In order to support designing with XAI and HITL methods, we will define an in-depth description of XAI/HITL-based interaction patterns based on existing taxonomies [8, 24, 49, 51, 67]. A mapping from interaction patterns to a set of implementation techniques will help us explore possible commonalities between design and implementation aspects. In order to address both limitations, our proposed design space will be informed by existing guidelines and frameworks related to the design and implementation of HAI interactions.

5.2 Envisioned Applications

We propose a design space which can support designers and AI practitioners to design and implement HAI interactions based on interaction primitives and patterns. More specifically, we envision a design space which can enable users to explore and choose between existing patterns and modify them towards new types of interactions. The design space will be developed as a prototyping tool by providing suggestions about the design and implementation aspects of existing and new patterns. Building on the existing list of patterns and actions (Tables 4, 5), an extendable collection of interaction patterns and their design/implementation choices will be used as design materials for more complex HAI interactions. A mapping from interaction patterns to common implementation aspects could support fast prototyping of HAI interactions, in the form of auto-generated code for basic model operations during HAI interactions, e.g., data operations, model predict, fit, etc.. A key aspect of the proposed space is to link design aspects with implementation choices for a given interaction concept. Towards this, we provide a short description of the implementation aspects of extracted patterns (Section 4.3), aiming to identify possible commonalities between patterns and implementation issues. Finally, we discuss how the proposed space can be informed by existing frameworks and guidelines.

5.2.1 Manifesting implementation concerns. A main motivation for the proposed design space is the need to bridge the gap between design and implementation choices. This can be achieved by characterizing the defined interaction patterns in terms of interface and implementation requirements. We envisage that developing a collection of interaction patterns as well-defined, named entities, can enable us to extract a list of common implementation issues attached to a given interaction pattern. Towards this, we provide an overview of the implementation aspects for the defined patterns, considering the different HAI interaction paradigms.

XAI-based interaction patterns. Based on the literature review and the unpacking process, we identify the following implementation aspects and challenges regarding XAI-based interaction patterns: (a) the selection of appropriate explanation strategies and methods considering the interaction goals [1, 8, 55], (b) the adaptation of explanations based on user's roles and characteristics (e.g., expertise, perception, etc.) [17, 41], and (c) the evaluation of XAI methods in terms of the interaction goal and user's behavior [18, 49, 63]. These can be related to the patterns found above:

- The `prediction-with-XAI` is used to help a non-technical user to understand if they can trust the model's prediction or not. This requires the model to communicate its accuracy rate along with a prediction, in an intuitive way (e.g., chart) in order to manage user's expectations about the model decisions.
- The `prediction_based-XAI` requires a proper explanation method considering the role of the user. For a domain expert user, it needs to provide appropriate information towards altering the model's decisions, while for a non-technical user, it considers the user's characteristics, e.g., cognitive load or preferences. The accuracy of the explanations is a key factor towards an effective interaction.
- The `prediction-parameters` is used to help the user identify the weaknesses of the model, e.g., prediction with low confidence. This requires the model to communicate its confidence values along with its predictions

for a set of input samples, towards a scrutable model. The user is able to explore and alter the model decisions, providing feedback for model updates.
- The `policy visualization` is used to communicate the model's policy in an online fashion. In terms of implementation, the model communicates and updates its policy (input-output) during the interactions. This requires an online policy update mechanism to enable the user get an understanding of how the model's performance changes over time.

HITL-based interaction patterns. The goal of HITL methods and approaches is to efficiently integrate the human user to the learning and decision making process of an AI system. Human users can participate in the model's development and deployment phases. The selection of appropriate iML/HITL methods and approaches depends on several aspects of the interaction, e.g., goal, user role and expertise, etc. For a given context, HITL-based patterns can be used to design interactions where the user is part of the decision making and learning process. We identify the following implementation challenges for HITL/iML-based interactions: (a) the selection of teaching strategies considering user roles and expertise [16, 19], (b) the design of intuitive and user-friendly feedback/control interfaces to ensure high-quality feedback [24, 28], and (c) the integration of feedback (models) to model updates and decision making [48, 50]. We can relate these to the patterns above as:

- `sample-annotation` is used to enable the user provide a label for a sample. A key decision relates to the quality of the user-provided data. For the interactive robot learning, the model considers the human label as the ground truth for the generated sample, based on which it performs the learning update (online training). User is able to provide training data implicitly during the interaction.
- `evaluative-advice` is used to enable a user to evaluate the model's performance. Evaluation feedback requires an interface for numerical feedback, as well as a reward shaping mechanism to integrate human feedback into model updates. User needs to be able to perceive and evaluate the model's predictions (policy) in an online manner.
- `informative-advice` is used to enable provide an alternate decision. Informative advice requires an interface for action selection, as well as a policy shaping mechanism to integrate human feedback into model updates. User needs to be able to perceive and modify the model's predictions (policy) in an online manner.
- `modify-mparams` is used as a control pattern which enables the user to alter the model's decisions by changing a model's parameter. In terms of implementation, the model should be able to dynamically alter its decisions based on the modified input. The model uses feedback only to alter its predictions for a new parameter and not to update its learning weights.

Hybrid Intelligence and Collaborative learning interaction patterns The goal of collaborative learning and hybrid intelligence interactions is to enable both humans and machines (AI systems) to learn from each other in a collaborative manner. Collaborative learning interactions can be designed by combining XAI-based and HITL-based interactions, enabling both users and AI models to exchange information while solving a task. Based on our unpacking process, we identify and discuss interaction patterns used in collaborative learning and hybrid intelligence interactions. Implementation challenges for hybrid intelligence and collaborative learning systems include, but not limited to: (a) identify the appropriate levels of human control and AI automation both for model learning and decision making [64, 85], (b) design systems to facilitate the interaction between explainable artificial and cognitive intelligence [74], and (c) develop methods for the adaptation of HI systems considering user needs and capabilities to enhance user's perception towards

improving the model's learning process [4]. Considering these, we identify the following challenges for the extracted patterns:

- The `turn_taking-evaluation` pattern is used to enable the user collaborate with a model in a turn-taking interaction. In terms of implementation, the model needs to capture the user's input during the interaction and generate a modified sketch. The user can control what the robot will capture, as well as when to provide feedback and terminate the interaction.
- The `candidate-samples` pattern is used to support the human labeling process in a semi-automated way. The user makes the final decisions based on the model's initial decisions. The model is being updated based on user's feedback. The model needs to make online updates to improve the selection of the candidate samples, which can affect the user's and thus model's performance.
- The `turn-taking_XAI` pattern enables both model and user to justify their decisions using explanations. The implementation of this pattern requires an interface where the user can modify the model's prediction and explanations. The type of explanations (visualization) depends on user characteristics, e.g., preferences, cognitive load.

Focusing on the implementation aspects of these patterns, we observe that each pattern can be linked to a set of technical aspects that need to be considered. We envision a library of patterns which can characterize a pattern in terms of the possible implementation choices to manifest common concerns and issues. In order to support both design and implementation choices, our proposed design space will be informed by existing frameworks and guidelines for HAI interactions.

5.2.2 Relation to existing frameworks and guidelines. In order to get insights about the formalization of the proposed design space, we briefly discuss how existing design guidelines and frameworks are related to our proposed approach. Our vision is a design space which can support AI designers to explore this space between design guidelines and implementation practices by enabling the collaboration of such frameworks and guidelines (Figure 10).

Fig. 10. Our proposed design space as a link between design guidelines and implementation choices.

For example, the Microsoft guidelines for Human-AI interactions [7] provide a set of pattern examples along with descriptions of interactions. These design guidelines can be used as a set of suggestions and pattern examples for the ideation phase of an HAI interaction system. For example, a guideline suggests helping the user understands what the system can do[3]. One of the suggested pattern examples for this given guideline is to use explanation patterns in order to enable users to gain insights into system capabilities *(XAI-based interaction patterns)*. Based on another guideline, the design should encourage granular feedback and enable the user to indicate their preferences[4]. A suggested pattern

[3]https://www.microsoft.com/en-us/haxtoolkit/guideline/make-clear-what-the-system-can-do/
[4]https://www.microsoft.com/en-us/haxtoolkit/guideline/encourage-granular-feedback/

for this guideline is to request explicit feedback on selected system outputs in order to assess the system and help it improve over time (HITL-based interactions). Our proposed design space could be informed by the provided guidelines and pattern examples in order to develop a collection of design patterns considering these guidelines. Working at a higher-level, the Assessment List for Trustworthy AI (ALTAI)[5] was developed for the assessment of AI systems in terms of seven requirements specified in the Ethics Guidelines for Trustworthy AI [5]. *Human Agency and Oversight* is one of the requirements and refers to the ability of human users to make informed decisions and to monitor and supervise the system. HITL-based interaction patterns can be used to integrate the user to the decision making and model learning process. Another ALTAI guideline is *Transparency* and refers to the ability of the data, system and AI models to be transparent and explainable to the user. XAI-based interaction patterns can describe different approaches to provide explanations to the user based on the role and intent in the interaction. Considering the assessment list and requirements, our proposed design space can provide support towards selecting appropriate patterns and interactions to comply with specific assessment items.

Apart from high-level guidelines, our proposed design space will be informed by technical and implementation frameworks for AI/ML models. Considering interactions with ML systems, implementation methods are required to enable (a) the communication of information between users and models and (b) the integration of user-provided data to the model learning (and decision making) mechanism. A classification of methods and approaches for interactive ML systems [51] considers the development lifecycle of ML systems and provides a list of implementation choices based on a given category of methods, including interactive learning and explainability. However, designing efficient HITL-based interaction patterns requires both the selection of appropriate design choices and learning methods for feedback integration. Focusing on detecting and mitigating bias in ML pipelines, FaiPrep is an open library which extracts dataflow representations to support fairness during model development [80]. Such representations can be used in our context to further characterize HAI interactions in terms of dataflows and model/data operations, e.g., fit, predict, update data. Finally, our proposed design space will be informed by existing approaches for design patterns for AI systems. Design patterns have been proposed for hybrid AI and reasoning systems, combining both data-driven and knowledge-driven AI models [70]. These hybrid AI patterns can specify the operations that take place during the interactions (e.g., fit, update, predict) and can provide guidelines for the selection of appropriate learning methods.

6 CONCLUSION

The design space developed here is aimed at the complex space between concepts and practice, between formality and accessibility. The interaction patterns developed here act as intermediate level knowledge for the design and understanding of HAI systems, giving a formal representation of the configurations used in existing work. Starting from a small set of interaction primitives and types to specify the communicated information between the interacting agents, we showed that the proposed primitives can be used to describe patterns of interactions from a range of systems, resulting in a collection of crisply defined actions and interaction patterns. We demonstrated that these patterns and actions are consistent with key HAI paradigms of HITL, XAI and hybrid intelligence, and that they can be used to explore and prototype a range of alternate interactions for a given situation.

This space has been built from theoretical ideas about communication between humans and models, based on ideas from agent communication languages and the semantic interaction framework for understanding human-model communication. It has then been developed and tested with examples of systems and interaction paradigms from the

[5]https://altai.insight-centre.org/

literature, demonstrating that it can meaningfully describe existing work. The representation language starts with data types and primitive communicative actions of providing and requesting information, and works up to high level conceptual activities — interaction patterns — that both capture common structures and describe the intent of the computational architectures. Extracting these patterns gives people of varying technicalities a common language to talk about what a particular system is doing, by building re-usable descriptions of the interactions taking place. This level of description allows for alternative design choices to be explored, while highlighting concerns that might arise and giving a framework for implementing the interactions. This provides a way to document existing practices, re-use well tested solutions and also speculate about new interaction possibilities through an exploration of the design space. Finally, through focussing on the interactive and communicative possibilities around models, the design space helps to shift thinking from a single user, single model, single purpose viewpoint to one where various stakeholders can carry out different kinds of interaction with a single model, creating a more ecosystemic view of human-model interactions.

REFERENCES

[1] Ajaya Adhikari, Edwin Wenink, Jasper van der Waa, Cornelis Bouter, Ioannis Tolios, and Stephan Raaijmakers. 2022. Towards FAIR Explainable AI: a standardized ontology for mapping XAI solutions to use cases, explanations, and AI systems. In *Proceedings of the 15th International Conference on PErvasive Technologies Related to Assistive Environments*. 562–568.

[2] Raheel Ahmad, Shahram Rahimi, and Bidyut Gupta. 2007. An Intelligence-Aware Process Calculus for Multi-Agent System Modeling. In *2007 International Conference on Integration of Knowledge Intensive Multi-Agent Systems*. IEEE, 210–215.

[3] Moamin Ahmed, Mohd Sharifuddin Ahmad, and Mohd Zaliman Mohd Yusoff. 2009. A review and development of agent communication language. *Electronic Journal of Computer Science and Information Technology* 1, 1 (2009).

[4] Zeynep Akata, Dan Balliet, Maarten De Rijke, Frank Dignum, Virginia Dignum, Guszti Eiben, Antske Fokkens, Davide Grossi, Koen Hindriks, Holger Hoos, et al. 2020. A research agenda for hybrid intelligence: augmenting human intellect with collaborative, adaptive, responsible, and explainable artificial intelligence. *Computer* 53, 08 (2020), 18–28.

[5] Pekka Ala-Pietilä, Yann Bonnet, Urs Bergmann, Maria Bielikova, Cecilia Bonefeld-Dahl, Wilhelm Bauer, Loubna Bouarfa, Raja Chatila, Mark Coeckelbergh, Virginia Dignum, et al. 2020. *The assessment list for trustworthy artificial intelligence (ALTAI)*. European Commission.

[6] Kars Alfrink, Ianus Keller, Gerd Kortuem, and Neelke Doorn. 2022. Contestable AI by Design: Towards a Framework. *Minds and Machines* (2022), 1–27.

[7] Saleema Amershi, Dan Weld, Mihaela Vorvoreanu, Adam Fourney, Besmira Nushi, Penny Collisson, Jina Suh, Shamsi Iqbal, Paul N Bennett, Kori Inkpen, et al. 2019. Guidelines for human-AI interaction. In *Proceedings of the 2019 chi conference on human factors in computing systems*. 1–13.

[8] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. 2020. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information fusion* 58 (2020), 82–115.

[9] Agathe Balayn, Panagiotis Soilis, Christoph Lofi, Jie Yang, and Alessandro Bozzon. 2021. What Do You Mean? Interpreting Image Classification with Crowdsourced Concept Extraction and Analysis. In *Proceedings of the Web Conference 2021*. ACM, Ljubljana Slovenia, 1937–1948. https://doi.org/10.1145/3442381.3450069

[10] Kyle J Behymer and John M Flach. 2016. From autonomous systems to sociotechnical systems: Designing effective collaborations. *She Ji: The Journal of Design, Economics, and Innovation* 2, 2 (2016), 105–114.

[11] Umang Bhatt, Javier Antorán, Yunfeng Zhang, Q Vera Liao, Prasanna Sattigeri, Riccardo Fogliato, Gabrielle Melançon, Ranganath Krishnan, Jason Stanley, Omesh Tickoo, et al. 2021. Uncertainty as a form of transparency: Measuring, communicating, and using uncertainty. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*. 401–413.

[12] Adam Bignold, Francisco Cruz, Richard Dazeley, Peter Vamplew, and Cameron Foale. 2022. Human engagement providing evaluative and informative advice for interactive reinforcement learning. *Neural Computing and Applications* (2022), 1–16.

[13] Abeba Birhane. 2021. Algorithmic injustice: a relational ethics approach. *Patterns* 2, 2 (2021), 100205.

[14] Luciano Cavalcante Siebert, Maria Luce Lupetti, Evgeni Aizenberg, Niek Beckers, Arkady Zgonnikov, Herman Veluwenkamp, David Abbink, Elisa Giaccardi, Geert-Jan Houben, Catholijn M Jonker, et al. 2022. Meaningful human control: Actionable properties for AI system development. *AI and Ethics* (2022), 1–15.

[15] Chengliang Chai and Guoliang Li. 2020. Human-in-the-loop Techniques in Machine Learning. *IEEE Data Eng. Bull.* 43, 3 (2020), 37–52.

[16] Valerie Chen, Umang Bhatt, Hoda Heidari, Adrian Weller, and Ameet Talwalkar. 2022. Perspectives on Incorporating Expert Feedback into Model Updates. *arXiv preprint arXiv:2205.06905* (2022).

[17] Michael Chromik and Andreas Butz. 2021. Human-XAI interaction: a review and design principles for explanation user interfaces. In *IFIP Conference on Human-Computer Interaction*. Springer, 619–640.

[18] Michael Chromik and Martin Schuessler. 2020. A Taxonomy for Human Subject Evaluation of Black-Box Explanations in XAI. *Exss-atec@ iui* 94 (2020).

[19] Yuchen Cui, Pallavi Koppol, Henny Admoni, Scott Niekum, Reid G Simmons, Aaron Steinfeld, and Tesca Fitzgerald. 2021. Understanding the Relationship between Interactions and Outcomes in Human-in-the-Loop Machine Learning.. In *IJCAI*. 4382–4391.

[20] Dominik Dellermann, Adrian Calma, Nikolaus Lipusch, Thorsten Weber, Sascha Weigel, and Philipp Ebel. 2021. The future of human-AI collaboration: a taxonomy of design knowledge for hybrid intelligence systems. *arXiv preprint arXiv:2105.03354* (2021).

[21] Mark d'Inverno, Michael Luck, Pablo Noriega, Juan A. Rodriguez-Aguilar, and Carles Sierra. 2012. Communicating Open Systems. *Artificial Intelligence* 186 (July 2012), 38–94. https://doi.org/10.1016/j.artint.2012.03.004

[22] Graham Dove and Anne-Laure Fayard. 2020. Monsters, Metaphors, and Machine Learning. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–17. https://doi.org/10.1145/3313831.3376275

[23] Graham Dove, Kim Halskov, Jodi Forlizzi, and John Zimmerman. 2017. UX design innovation: Challenges for working with machine learning as a design material. In *Proceedings of the 2017 chi conference on human factors in computing systems*. 278–288.

[24] John J Dudley and Per Ola Kristensson. 2018. A review of user interface design for interactive machine learning. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 8, 2 (2018), 1–37.

[25] Lilian Edwards and Michael Veale. 2017. *Slave to the Algorithm? Why a 'right to an Explanation' Is Probably Not the Remedy You Are Looking For.* Preprint. LawArXiv. https://doi.org/10.31228/osf.io/97upg

[26] Upol Ehsan and Mark O Riedl. 2020. Human-centered explainable ai: Towards a reflective sociotechnical approach. In *International Conference on Human-Computer Interaction*. Springer, 449–466.

[27] Therese Enarsson, Lena Enqvist, and Markus Naarttijärvi. 2022. Approaching the human in the loop–legal perspectives on hybrid human/algorithmic decision-making in three contexts. *Information & Communications Technology Law* 31, 1 (2022), 123–153.

[28] Alex Endert, Patrick Fiaux, and Chris North. 2012. Semantic interaction for sensemaking: inferring analytical reasoning for model steering. *IEEE Transactions on Visualization and Computer Graphics* 18, 12 (2012), 2879–2888.

[29] Tim Finin, Richard Fritzson, Don McKay, and Robin McEntire. 1994. KQML as an agent communication language. In *Proceedings of the third international conference on Information and knowledge management*. 456–463.

[30] Elisa Giaccardi and Johan Redström. 2020. Technology and more-than-human design. *Design Issues* 36, 4 (2020), 33–44.

[31] Marco Gillies. 2019. Understanding the role of interactive machine learning in movement interaction design. *ACM Transactions on Computer-Human Interaction (TOCHI)* 26, 1 (2019), 1–34.

[32] Imke Grabe, Miguel González-Duque, Sebastian Risi, and Jichen Zhu. 2022. Towards a Framework for Human-AI Interaction Patterns in Co-Creative GAN Applications. (2022).

[33] Tor Grønsund and Margunn Aanestad. 2020. Augmenting the algorithm: Emerging human-in-the-loop work configurations. *The Journal of Strategic Information Systems* 29, 2 (2020), 101614.

[34] Lijie Guo, Elizabeth M Daly, Oznur Alkan, Massimiliano Mattetti, Owen Cornec, and Bart Knijnenburg. 2022. Building Trust in Interactive Machine Learning via User Contributed Interpretable Rules. In *27th International Conference on Intelligent User Interfaces*. 537–548.

[35] Alexander Heimerl, Tobias Baur, Florian Lingenfelser, Johannes Wagner, and Elisabeth André. 2019. NOVA-a tool for eXplainable Cooperative Machine Learning. In *2019 8th International Conference on Affective Computing and Intelligent Interaction (ACII)*. IEEE, 109–115.

[36] Tad Hirsch, Kritzia Merced, Shrikanth Narayanan, Zac E Imel, and David C Atkins. 2017. Designing contestability: Interaction design, machine learning, and mental health. In *Proceedings of the 2017 Conference on Designing Interactive Systems*. 95–99.

[37] Lars Erik Holmquist. 2017. Intelligence on tap: artificial intelligence as a new design material. *interactions* 24, 4 (2017), 28–33.

[38] Pranav Khadpe, Ranjay Krishna, Li Fei-Fei, Jeffrey T. Hancock, and Michael S. Bernstein. 2020. Conceptual Metaphors Impact Perceptions of Human-AI Collaboration. *Proceedings of the ACM on Human-Computer Interaction* 4, CSCW2 (2020), 1–26.

[39] Bongjun Kim and Bryan Pardo. 2018. A human-in-the-loop system for sound event detection and annotation. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 8, 2 (2018), 1–23.

[40] Rafal Kocielnik, Saleema Amershi, and Paul N Bennett. 2019. Will you accept an imperfect ai? exploring designs for adjusting end-user expectations of ai systems. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–14.

[41] Q Vera Liao, Daniel Gruen, and Sarah Miller. 2020. Questioning the AI: informing design practices for explainable AI user experiences. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–15.

[42] Q Vera Liao and Kush R Varshney. 2021. Human-centered explainable ai (xai): From algorithms to user experiences. *arXiv preprint arXiv:2110.10790* (2021).

[43] Yuyu Lin, Jiahao Guo, Yang Chen, Cheng Yao, and Fangtian Ying. 2020. It is your turn: collaborative ideation with a co-creative robot through sketch. In *Proceedings of the 2020 CHI conference on human factors in computing systems*. 1–14.

[44] Joseph Lindley, Haider Ali Akmal, Franziska Pilling, and Paul Coulton. 2020. Researching AI legibility through design. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–13.

[45] Michal Luria. 2018. Designing Robot Personality Based on Fictional Sidekick Characters. In *Companion of the 2018 ACM/IEEE International Conference on Human-Robot Interaction (HRI '18)*. Association for Computing Machinery, New York, NY, USA, 307–308. https://doi.org/10.1145/3173386.3176912

[46] Benedikt Maettig and Hermann Foot. 2020. Approach to improving training of human workers in industrial applications through the use of intelligence augmentation and human-in-the-loop. In *2020 15th International Conference on Computer Science & Education (ICCSE)*. IEEE, 283–288.

[47] Millecamp Martijn, Cristina Conati, and Katrien Verbert. 2022. "Knowing me, knowing you": personalized explanations for a music recommender system. *User Modeling and User-Adapted Interaction* 32, 1 (2022), 215–252.

[48] Chris J Michael, Dina Acklin, and Jaelle Scheuerman. 2020. On interactive machine learning and the potential of cognitive feedback. *arXiv preprint arXiv:2003.10365* (2020).

[49] Sina Mohseni, Niloofar Zarei, and Eric D Ragan. 2021. A multidisciplinary survey and framework for design and evaluation of explainable AI systems. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 11, 3-4 (2021), 1–45.

[50] Eduardo Mosqueira-Rey, Elena Hernández-Pereira, David Alonso-Ríos, José Bobes-Bascarán, and Ángel Fernández-Leal. 2022. Human-in-the-loop machine learning: a state of the art. *Artificial Intelligence Review* (2022), 1–50.

[51] Eduardo Mosqueira-Rey, Elena Hernández Pereira, David Alonso-Ríos, and José Bobes-Bascarán. 2022. A classification and review of tools for developing and interacting with machine learning systems. In *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*. 1092–1101.

[52] Dave Murray-Rust, Iohanna Nicenboim, and Dan Lockton. 2022. Metaphors for Designers Working with AI. In *DRS Biennial Conference Series*. https://doi.org/10.21606/drs.2022.667

[53] Dave Murray-Rust, Petros Papapanagiotou, and Dave Robertson. 2015. Softening electronic institutions to support natural interaction. *Human Computation* 2, 2 (2015).

[54] Mario Nadj, Merlin Knaeble, Maximilian Xiling Li, and Alexander Maedche. 2020. Power to the oracle? design principles for interactive labeling systems in machine learning. *KI-Künstliche Intelligenz* 34, 2 (2020), 131–142.

[55] Ingrid Nunes and Dietmar Jannach. 2017. A systematic review and taxonomy of explanations in decision support and recommender systems. *User Modeling and User-Adapted Interaction* 27, 3 (2017), 393–444.

[56] Paul D O'Brien and Richard C Nicol. 1998. FIPA—towards a standard for software agents. *BT Technology Journal* 16, 3 (1998), 51–59.

[57] Kieron O'Hara. 2020. Explainable AI and the Philosophy and Practice of Explanation. *Computer Law & Security Review* 39 (Nov. 2020), 105474. https://doi.org/10.1016/j.clsr.2020.105474

[58] Dorian Peters, Karina Vold, Diana Robinson, and Rafael A Calvo. 2020. Responsible AI—two frameworks for ethical design practice. *IEEE Transactions on Technology and Society* 1, 1 (2020), 34–47.

[59] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-shot text-to-image generation. In *International Conference on Machine Learning*. PMLR, 8821–8831.

[60] Jeba Rezwana and Mary Lou Maher. 2022. Designing Creative AI Partners with COFI: A Framework for Modeling Interaction in Human-AI Co-Creative Systems. *ACM Transactions on Computer-Human Interaction* (2022).

[61] David Robertson. 2005. A lightweight coordination calculus for agent systems. In *Declarative Agent Languages and Technologies II: Second International Workshop, DALT 2004, New York, NY, USA, July 19, 2004, Revised Selected Papers 2*. Springer, 183–197.

[62] Tjeerd AJ Schoonderwoerd, Emma M van Zoelen, Karel van den Bosch, and Mark A Neerincx. 2022. Design patterns for human-AI co-learning: A wizard-of-Oz evaluation in an urban-search-and-rescue task. *International Journal of Human-Computer Studies* 164 (2022), 102831.

[63] Gesina Schwalbe and Bettina Finzel. 2021. A Comprehensive Taxonomy for Explainable Artificial Intelligence: A Systematic Survey of Surveys on Methods and Concepts. *arXiv e-prints* (2021), arXiv–2105.

[64] Ben Shneiderman. 2020. Human-centered artificial intelligence: Reliable, safe & trustworthy. *International Journal of Human–Computer Interaction* 36, 6 (2020), 495–504.

[65] Fabian Sperrle, Mennatallah El-Assady, Grace Guo, Duen Horng Chau, Alex Endert, and Daniel Keim. 2020. Should we trust (x) AI? Design dimensions for structured experimental evaluations. *arXiv preprint arXiv:2009.06433* (2020).

[66] Hariharan Subramonyam, Colleen Seifert, and Eytan Adar. 2021. Towards a process model for co-creating AI experiences. In *Designing Interactive Systems Conference 2021*. 1529–1543.

[67] Stefano Teso, Öznur Alkan, Wolfgang Stammer, and Elizabeth Daly. 2023. Leveraging explanations in interactive machine learning: An overview. *Frontiers in Artificial Intelligence* 6 (2023). https://doi.org/10.3389/frai.2023.1066049

[68] Mike Treanor, Alexander Zook, Mirjam P Eladhari, Julian Togelius, Gillian Smith, Michael Cook, Tommy Thompson, Brian Magerko, John Levine, and Adam Smith. 2015. AI-based game design patterns. (2015).

[69] Konstantinos Tsiakas, Maher Abujelala, and Fillia Makedon. 2018. Task engagement as personalization feedback for socially-assistive robots and cognitive training. *Technologies* 6, 2 (2018), 49.

[70] Michael van Bekkum, Maaike de Boer, Frank van Harmelen, André Meyer-Vitali, and Annette ten Teije. 2021. Modular design patterns for hybrid learning and reasoning systems. *Applied Intelligence* 51, 9 (2021), 6528–6546.

[71] Niels van Berkel, Mikael B Skov, and Jesper Kjeldskov. 2021. Human-AI interaction: intermittent, continuous, and proactive. *Interactions* 28, 6 (2021), 67–71.

[72] Danding Wang, Qian Yang, Ashraf Abdul, and Brian Y Lim. 2019. Designing theory-driven user-centric explainable AI. In *Proceedings of the 2019 CHI conference on human factors in computing systems*. 1–15.

[73] Katharina Weitz, Dominik Schiller, Ruben Schlagowski, Tobias Huber, and Elisabeth André. 2021. "Let me explain!": exploring the potential of virtual agents in explainable AI interaction design. *Journal on Multimodal User Interfaces* 15, 2 (2021), 87–98.

[74] John Wenskovitch and Chris North. 2020. Interactive Artificial Intelligence: Designing for the" Two Black Boxes" Problem. *Computer* 53, 8 (2020), 29–39.

[75] Christina Wiethof and E Bittner. 2021. Hybrid intelligence-combining the human in the loop with the computer in the loop: a systematic literature review. In *Forty-Second International Conference on Information Systems, Austin*.

[76] Xingjiao Wu, Luwei Xiao, Yixuan Sun, Junhang Zhang, Tianlong Ma, and Liang He. 2022. A survey of human-in-the-loop for machine learning. *Future Generation Computer Systems* (2022).

[77] Wei Xu. 2019. Toward human-centered AI: a perspective from human-computer interaction. *interactions* 26, 4 (2019), 42–46.

[78] Wei Xu, Marvin J Dainoff, Liezhong Ge, and Zaifeng Gao. 2022. Transitioning to human interaction with AI systems: New challenges and opportunities for HCI professionals to enable human-centered AI. *International Journal of Human–Computer Interaction* (2022), 1–25.

[79] Jinyu Yang and Bo Zhang. 2019. Artificial intelligence in intelligent tutoring robots: A systematic review and design guidelines. *Applied Sciences* 9, 10 (2019), 2078.

[80] Ke Yang, Biao Huang, Julia Stoyanovich, and Sebastian Schelter. 2020. Fairness-Aware Instrumentation of Preprocessing˜ Pipelines for Machine Learning. In *Workshop on Human-In-the-Loop Data Analytics (HILDA'20)*.

[81] Qian Yang, Nikola Banovic, and John Zimmerman. 2018. Mapping machine learning advances from hci research to reveal starting places for design innovation. In *Proceedings of the 2018 CHI conference on human factors in computing systems*. 1–11.

[82] Qian Yang, Aaron Steinfeld, Carolyn Rosé, and John Zimmerman. 2020. Re-examining whether, why, and how human-AI interaction is uniquely difficult to design. In *Proceedings of the 2020 chi conference on human factors in computing systems*. 1–13.

[83] Chuang Yu and Adriana Tapus. 2019. Interactive robot learning for multimodal emotion recognition. In *International Conference on Social Robotics*. Springer, 633–642.

[84] Mireia Yurrita, Dave Murray-Rust, Agathe Balayn, and Alessandro Bozzon. 2022. Towards a Multi-Stakeholder Value-Based Assessment Framework for Algorithmic Systems. In *2022 ACM Conference on Fairness, Accountability, and Transparency (FAccT '22)*. Association for Computing Machinery, New York, NY, USA, 535–563. https://doi.org/10.1145/3531146.3533118

[85] Zelun Tony Zhang, Yuanting Liu, and Heinrich Hussmann. 2021. Forward reasoning decision support: toward a more complete view of the human-AI interaction design space. In *CHItaly 2021: 14th Biannual Conference of the Italian SIGCHI Chapter*. 1–5.

[86] Xiaofei Zhou, Jessica Van Brummelen, and Phoebe Lin. 2020. Designing AI learning experiences for K-12: emerging works, future opportunities and a design framework. *arXiv preprint arXiv:2009.10228* (2020).

A APPENDIX

A.1 Unpacking of human-AI interactions: use cases

This section includes a list of uses cases to demonstrate our proposed unpacking approach. Based on the selected use cases, we extract actions and patterns for different interaction concepts, including XAI-based and HITL interactions. More specifically, the list includes an interactive sound annotation system [39], an explainable active learning tool for image classification [35], a human-robot interaction for collaborative sketching [43], an music recommendation system using personalized explanations [47], an interactive meeting scheduling assistant [40] and an interactive ML approach for game-based collaboration[34]. For each use case, we describe the interactions (unpacking) and we define the actions and patterns based on the defined primitives.

A.1.1 Human-in-the-loop sound event detection and annotation (Figure 11). The proposed system integrates a user interface for interactive sound event detection and annotation. The user sets a target sound event by selecting or uploading a sound segment which includes the target sound (e.g., door knocking). With this interaction, the user sets the possible model output classes: positive when the segment includes the sound target and negative otherwise. The model selects and highlights segments similar to the positive sample (user's input) and asks the user to classify the segments. The user can either provide a label for a segment (target or not) or adjust the segment boundaries, if the segment does not include the full sound. The model utilizes user's feedback (re-labeling/re-segmentation) to update its model parameters in an iterative and interactive manner.

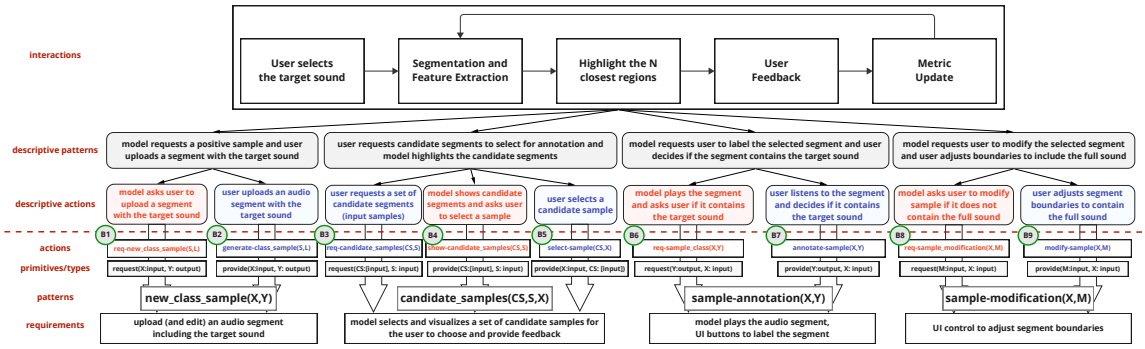

Fig. 11. Unpacking an interactive sound annotation interaction [39] into interaction primitives and patterns.

We identify three types of interactions: *(a) positive sample:* the model asks the user for a positive example (segment that includes the target sound) - user provides a positive sample, *(b) select candidate sample:* user requests a set of candidate segments – model highlights the candidate samples – user selects a candidate segment, *(c) label sample:* model plays segment and asks user if it contains the target sound – user responds by labeling the segment, and *(d) modify sample:* model plays segment and asks user if it contains the full sound – user adjusts the boundaries of the segment until full sound is included. During these interactions, user and model exchange information through positive samples, visualization and selection of candidate segments and their current labels (highlighted), playing/listening to segments, and re-labeling or re-segmenting selected samples. Considering the design and implementation approach, the proposed approach focuses on two aspects: (a) the design of a human-friendly interface which can provide interactivity and feedback control, and (b) the iterative labeling and model training approach, based on which the model dynamically

recalculates the model parameters (weights) based on user feedback. In terms of evaluation metrics, both user-based (interaction overhead) and algorithm-based (model accuracy) measurements were considered. Their analysis indicates that minimizing the interaction overhead (through design) can maximize machine performance and speed. Table 6 shows the messages and action definitions.

	Message	Action definition
B1	<model→user,**req-new_class_sample(S,L)**, [X:uploadBtn,targetSound;Y:positiveLabel]>	req-new_class_sample(S,L)≡ request(X:input.raw_data,Y:output.label) ← create(S),map(S,L)
B2	<user→model,**generate-class_sample(S,L)**, [X:uploadTargetSound;Y:positiveLabel]>	generate-class_sample(S,L) ≡ provide(S:input.raw_data,L:output.label) ←create(S),map(S,L)
B3	<user→model,**req-candidate_samples(CS,S)**, [CS:similarSegs,highlightSeg;S:posSample]>	req-candidate_samples(CS,S)≡ request(CS:[input.raw_data],S:input.raw_data) ←select(CS),map(CS,S)
B4	<model→user,**show-candidate_samples(CS,S)**, [CS:similarSegs,highlightSeg;S:posSample]>	show-candidate_samples(CS,S)≡ provide(CS:[input.raw_data],S:input.raw_data) ←select(CS),map(CS,S)
B5	<user→model,**select-sample(X,CS)**, [X:selSegment;CS:highlightSeg]>	select-sample(X,CS) ≡ provide(X:input.raw_data,CS:[input.raw_data]) ←select(X,CS)
B6	<model→user,**req-sample_class(X,Y)**, [X:playSegment;Y:isTargetSound,labelBtn]>	req-sample_class(X,Y) ≡ request(Y:output.label,X:input.raw_data) ← map(X,Y)
B7	<user→model,**annotate-sample(X,Y)**, [X:selSegment; Y:isTargetSound,labelBtn]>	annotate-sample(X,Y)≡ provide(Y:output.label,X:input.raw_data) ←map(X,Y)
B8	<model→user,**req-modified-sample(X,M)**, [X:selSegment;M:modifySegment]>	req-modified-sample(X,M)≡ request(M:input.raw_data,X:input.raw_data) ←modify(X,M)
B9	<model→user,**modify-sample(X,M)**, [X:selSegment;M:modifySegment]>	modify-sample(X,M)≡ provide(M:input.raw_data,X:input.raw_data) ←modify(X,M)

Table 6. Messages and action definitions for the interactive sound annotation system interactions

pattern	actions/messages	description
new_class_sample	req-new-class_sample(S,L)	model asks user for a (positive) class sample
	generate-class_sample(S,L)	user provides a (positive) class sample
candidate_samples	req-candidate_samples(CS,S)	user asks for candidate (similar) samples
	show-candidate_samples(CS,S)	model provides a set of candidate samples
	select-sample(X,CS)	user selects a candidate sample
sample-annotation	req-sample_class(X,Y)	model asks user to provide a label for the input
	annotate-sample(X,Y)	user provides the correct label for the input
sample-modification	req-modified_sample(X,M)	model asks user to modify the sample (if needed)
	modify-sample(X,M)	user modifies the selected sample

Table 7. Interaction patterns for the interactive sound annotation system

Based on these definitions, we define the following patterns (Table 7): [B1–B2] new_class_sample, [B3–B5] candidate_samples, [B5–B7] sample-annotation, and [B8–B9] sample-modification. User and model interact through input (segments) and output (labels) in an interactive manner in order to annotate all candidate samples. More specifically, the first pattern is required to set the target class which represents samples which include a

target sound. This is achieved by asking the user to provide a segment which includes the required sound (positive sample). During the second pattern, the model uses the positive sample to identify and visualize a list of candidate inputs for the user to label. The third pattern describes how user provides feedback to the model by listening and annotating the selected segment. The fourth pattern describes the modification of a sample (input) by adjusting the boundaries of the selected segment.

A.1.2 Interactive RL and human trainer engagement (Figure 12). The proposed system integrates human-provided feedback to an RL agent to improve its performance while executing the Mountain Car task. More specifically, the RL agent visualizes the current state and the selected action based on the model's policy (input-output pair) and allows the user to provide two types of feedback (evaluative/informative) in order to complete the task. Evaluative advice assesses the past performance of an agent and it is provided in the form of a reward, while informative advice supplements future decision-making and it is provided as an intervention - modified action. The goal of the interaction is to utilize human advice and maximize model's performance.

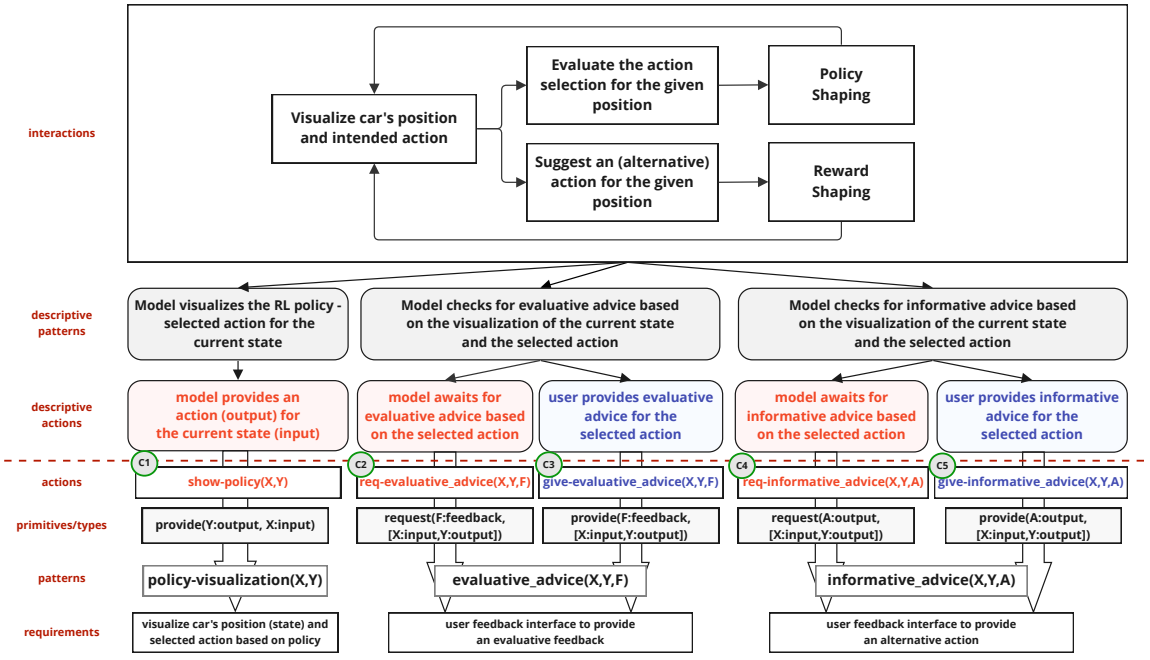

Fig. 12. Unpacking an interactive RL interaction into interaction patterns. Image adapted from [12]

We identify the following interactions: ***(a) policy visualization:*** model visualizes its policy as a selected action for the current state (input-output pair), ***(b) informative advice:*** model queries the user for informative advice – user provides an alternate action (output) for the current state-action pair, and ***(c)* evaluative advice:** model queries the user for evaluative advice – user provides a reward (feedback) for the current state-action pair. Interactions take place in the form of a transparent interactive RL policy (visualization) and human-feedback for advice or evaluation (through keyboard input). The goal of the user study is to measure human engagement and model performance for the two

types of feedback. In terms of the interaction design, human users are autonomous in terms of when they can provide feedback. According to the type of feedback, the model integrates it to the learning mechanism in different ways. If the user provides informative advice (learning from guidance), feedback is integrated to the learning mechanism through policy shaping, while for evaluative advice (learning from feedback), the RL agent uses reward shaping. The system uses an interface to visualize the RL policy and the task execution. Both types of advice are provided through a keyboard input, specific for each type. The design of the interactions plays an essential role in both model performance and human engagement. An important aspect to consider is the human feedback quality and consistency. The design challenge is to maintain user's engagement and performance considering user's perception. According to the analysis, users who provided informative advice were more engaged and accurate, which may be linked to how users perceive the different advice methods. Table 8 shows the messages and action definitions.

	Message	Action definition
C1	<model→user,**show-policy(X,Y)**, [X:CarPosition;Y:selectedAction]>	show-policy(X,Y)≡ provide([X:input.state,Y:output.action])) ← select(Y),map(X,Y)
C2	<model→user,**req-informative_advice(X,Y,A)**, [X:CarPosition;Y:selectedAction;A:keyboardAction]>	req-informative_advice(X,Y,A)≡ request(A:output.action,[X:input.state,Y:output.action]) ←modify(Y,A), map(X,A)
C3	<user→model,**give-informative_advice(X,Y,A)**, [X:CarPosition;Y:selectedAction;A:keyboardAction]>	give-evaluative_advice(X,Y,A)≡ provide(A:output.action,[X:input.state,Y:output.action]) ←modify(Y,A), map(X,A)
C4	<model→user,**req-evaluative_advice(X,Y,F)**, [X:CarPosition;Y:selectedAction;F:keyboardFeedback]>	req-evaluative_advice(X,Y,F)≡ request(F:feedback.eval,[X:input.state,Y:output.action]) ←select(F), map(X,A)
C5	<user→model,**give-evaluative_advice(X,Y,A)**, [X:CarPosition;Y:selectedAction;A:keyboardAction]>	give-evaluative_advice(X,Y,A)≡ provide(F:feedback.eval,[X:input.state,Y:output.action]) ←select(F),map(X,A)

Table 8. Messages and action definitions for the interactive RL interactions

Based on these definitions, we define the following patterns (Table 9): `[C1] policy-visualization`, `[C2-C3] informative_advice`, and `evaluative_advice`. The interaction starts with the policy visualization and a human teaching method. For both patterns, the RL agent communicates its policy by visualizing the current state and the selected action. Based on the teaching method, there are two different types of interactions between the human trainer and the model: informative advice in the form of an alternate action/guidance and evaluative advice as a feedback/reward. Each patterns requires an appropriate model update mechanism for online learning. For the evaluation pattern, the user can evaluate the policy by providing evaluative feedback (reward shaping), while for the informative pattern, the user can suggest an action as informative advice (policy shaping).

pattern	actions/messages	description
policy-visualization	`show-policy(X,Y)`	model visualizes action for current state
informative_advice	`req-informative_advice(X,Y,A)`	model checks if user provided advice
	`give-informative_advice(X,Y,A)`	user provides informative advice (action)
evaluative_advice	`req-evaluative_advice(X,Y,F)`	model checks if user provided feedback
	`give-evaluative_advice(X,Y,F)`	user provides evaluative feedback (reward)

Table 9. Interaction Patterns and actions for interactive RL.

A.1.3 Explainable active learning for collaborative emotion labeling (Figure 13). This use case is based on an application of a multimodal annotation tool, called NOVA. The use case describes a collaborative annotation task for emotion recognition. The proposed tool includes XAI functionalities (transparency and visualizations) which aim to enhance user's decision making and trust to the system. More specifically, the proposed annotation system follows an active learning approach to select which samples should be labeled by the user by visualizing the model's predictions confidence for these samples. The user can choose any sample and re-label it. Moreover, an XAI method (LIME visualization) is used to support user's decision making (emotion detection) through saliency maps; visualizations of important visual features for the selected frame classification.

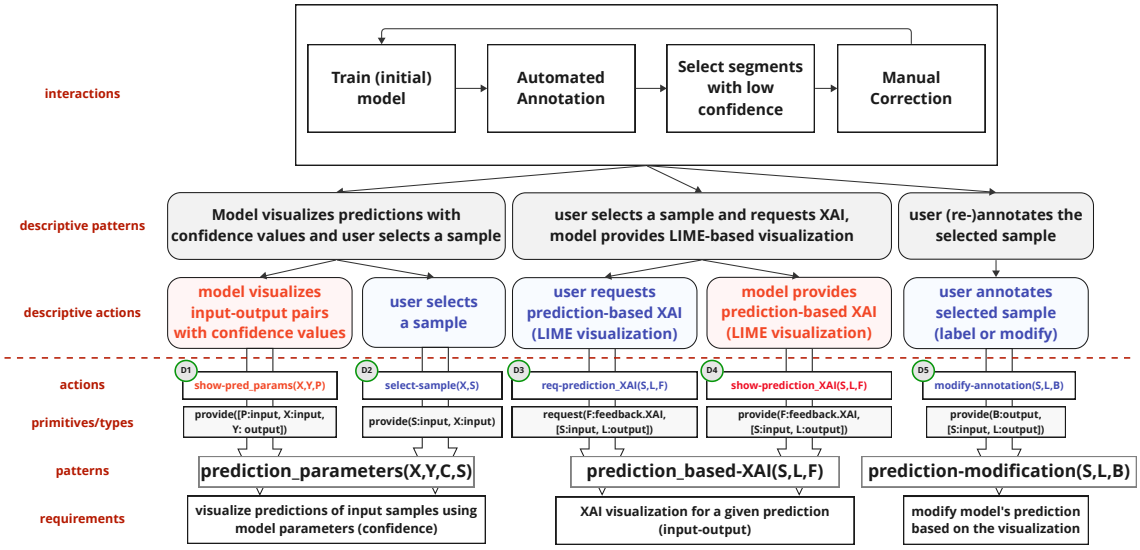

Fig. 13. Unpacking an explainable active learning interaction into patterns. Image adapted from [35]

We identify the following interaction patterns: (a) model provides input-output pairs with confidence values and visualizations, and (b) user selects and visualizes an input-output pair and updates it. These patterns can be further described as: (a1) model provides input, (a2) model provides output and (a3) model provides XAI-based feedback (confidence-based visualization), and (b1) user provides input (selection), (b2) user provides output (selection), (b3) user provides output (edit). The proposed system follows an explainable semi-supervised active learning approach. One of the main challenges of active learning is to identify the appropriate queries (data points) to ask for user labeling. The proposed approach aims to improve model performance through interactive labeling. Considering both design and implementation aspects, the system utilizes a user interface for model transparency and visual explanations, and integrates the human-in-the-loop to facilitate the active learning process, by identifying which data should be (re-)labeled. Both design and implementation choices are made to satisfy such requirements. Transparency and explainability are used to support user's decision making to refine a model through design features. The model guides the user to improve its performance for low-confidence predictions (active learning), while additional explanations (LIME) can be requested to further support user's annotation task. Based on these, we define the following actions (Table 10):

	Message	Action definition
D1	<model→user,**show-prediction_params(X,Y,P)**, [X:frames;Y:emotionLabels;P:confValues]>	show-prediction_params(X,Y,P)≡ provide([X:[input.raw_data],[Y:output.label],[P:input. model_params]]))← map(X,Y,P)
D2	<user→model,**select-sample(S,X)**, [X:frames;S:selFrame]>	select-sample(S,X)≡ provide(S:input.raw_data,X:[input.raw_data]) ←select(S,X)
D3	<user→model,**modify-prediction(S,L,A)**, [S:selFrame;L:emotionLabel;A:newLabel]>	modify-prediction(S,L,A)≡ provide(A:output.label,[S:input.raw_data,L:output.label]) ←modify(L,A), map(X,A)
D4	<model→user,**req-prediction_XAI(S,L,F)**, [S:selFrame;L:emotionLabel;F:LIMEVisualization]>	req-prediction_XAI(S,L,F)≡ request(F:feedback.XAI,[S:input.raw_data,L:output.label]) ←map(F,S,L)
D5	<model→user,**show-prediction_XAI(S,L,F)**, [S:selFrame;L:emotionLabel;F:LIMEVisualization]>	show-prediction_XAI(S,L,F)≡ provide(F:feedback.XAI,[S:input.raw_data,L:output.label]) ←map(F,S,L)

Table 10. Messages and action definitions for the active learning emotion recognition

We define the following patterns (Table 11): [D1–D2] prediction_parameters, [D3] prediction-modification, and [D4–D5] prediction-based_XAI. The first pattern describes the sample selection process, where the model supports the user to select the appropriate samples to annotate through visualizing the model confidence for its predictions. This approach is part of the active learning process based on which a set of candidate samples is selected for annotation. In this case, model instances with low confidence are presented to the user. The second pattern describes human labeling through a prediction modification approach. The third pattern describes a user request for XAI of a selected sample, where the model provides a LIME-based visualization for the user to understand the current prediction and modify it if needed. Local explanations are provided to the user to support their decision making through local interpretability. The output of LIME is a visualization of explanations representing the contribution of each feature to the prediction of the current frame. These patterns (and their actions) can be combined to design the interactions, e.g., the user can explore and select a sample based on the visualizations and either annotate it or request sample-based explanations.

pattern	actions/messages	description
prediction_parameters	show-prediction_params(X,Y,P)	parameter-based sample visualization
	select-sample(S,X)	user selects a sample from list
prediction-modification	modify-prediction(S,L,A)	user annotates/modifies an annotation
prediction-based_XAI	req-prediction_XAI(S,L,F)	user request XAI for selected sample
	show-prediction_XAI(S,L,F)	model provides sample-based XAI

Table 11. Interaction patterns and actions for the active learning emotion recognition

A.1.4 Human-robot collaborative sketching (Figure 14). The proposed system describes a collaborative sketching approach between a user an a robot (Cobbie). More specifically, the proposed system utilizes the mechanism of conceptual shift to support human-AI co-creation and collaborative sketch ideation. Based on this approach, the user initiates the interaction by sketching an image on paper. Once finished, the user gives the pen to the robot, which captures and analyzes the user's sketch. Based on its analysis, it generates a new sketch on paper. The user can pause the robot and provide an evaluative feedback for the robot's drawing. If feedback is negative, the robot starts drawing a new sketch until new feedback is received. If feedback is positive, the user draws a new sketch by combining the two sketches. The robot utilizes the provided feedback to adjust its model in order to provide more useful ideas to the user.

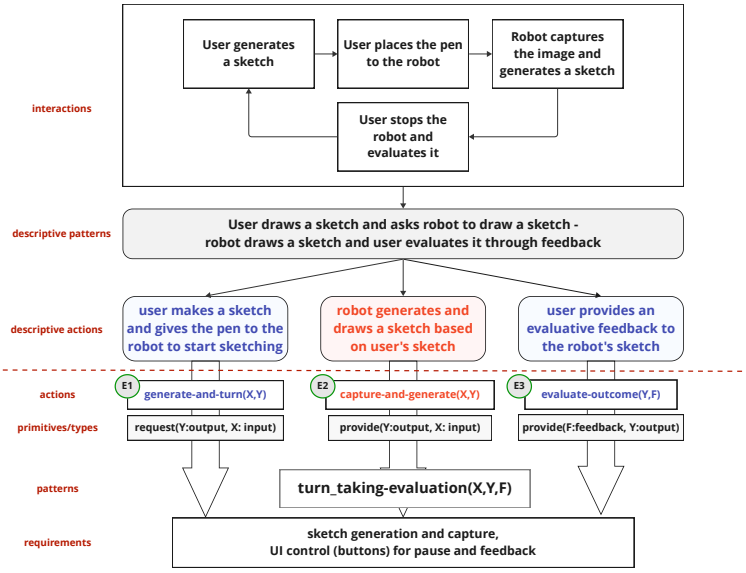

Fig. 14. Unpacking a human-robot collaborative sketching interaction into patterns. Image adapted from [43]

	Message	Action definition
E1	<model→user,**generate-and-turn(X,Y)**, [X:userSketch;Y:onPenClipper,robotSketch]>	generate-and-turn(X,Y)≡ request(Y:input.raw_data,C:output.raw_data)) ←create(X), create(Y), map(X,Y)
E2	<user→model,**capture-and-generate(X,Y)**, [X:userSketch; Y:robotSketch]>	capture-and-generate(X,Y)≡ provide(Y:input.raw_data,C:output.raw_data)) ←create(Y),map(X,Y)
E3	<user→model,**evaluate-outcome(Y,F)**, [Y:robotSketch;F:pauseBtn,feedbackBtn]>	evaluate-outcome(Y,F)≡ provide(F:feedback.eval,Y:output.raw_data) ←select(F), map(Y,F)

Table 12. Messages and action definitions for the collaborative sketching interactions.

The interaction takes place as a turn taking sketching-based interaction, where user and model sketch a drawing considering previous drawing (co-ideation). In terms of the types of communicated information, user and robot communicate through drawing sketches, giving the pen to the robot and providing feedback to the robot through buttons. In terms of implementation aspects, the robot deploys an RNN-based recognizer to capture and classify the user's input and an adapted version of the RNN-sketch model to generate a new image based on user's input. User's feedback is used to update the network weights, and thus In terms of interaction design, the user is the dominant member of the co-creation session. The user can determine when Cobbie should start drawing an image, by placing the pen to the robot's clipper. The user can also select the robot's position (what to capture and where to draw) as well as the pen strokes. Users can use the on-platform buttons to pause and resume robot's drawing and provide feedback. Robot expressive movements and sounds indicate that the robot has successfully received a command (button pressed). In terms or model performance, the deployed AI models (image classification and rnn-sketch) are well-performing models. The goal of the interaction is to support user's creative thinking and ideation processes. The model provides

appropriate outputs (not the most accurate) in order to facilitate this co-ideation process. Table 12 shows the message and actions definitions for this interaction.

pattern	actions	description
turn_taking-evaluation	`generate-and-turn(X,Y)`	user asks robot to sketch based on the drawing
	`capture-and-generate(X,Y)`	robot captures and generates new sketch
	`evaluate-outcome(Y,F)`	user pauses robot and provides feedback

Table 13. Interaction patterns and actions for the collaborative robot sketching.

Based on this interaction, we can define `[E]:turn_taking-evaluation` as a user-driven control and evaluation pattern, where the user can control and evaluate the collaboration with the model (Table 13). Based on this pattern, the user initiates the interaction by drawing a sketch and giving the pen to the robot for idea generation. The goal of the robot is to support user's ideation process by generating creative and diverse sketches. This is achieved through object detection (image classification) and conceptual shift (sketc-rnn). The user evaluates the robot's sketches (model output) using the feedback buttons. This interaction pattern can describe an explorative collaborative learning process, where both user and system aim to explore and identify new insights through their interaction.

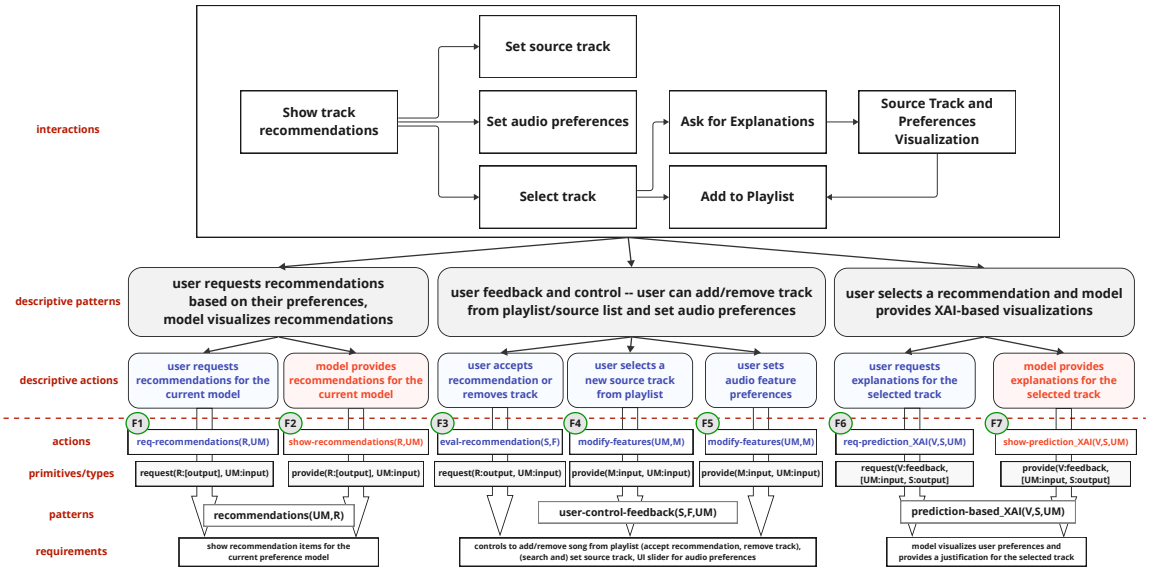

Fig. 15. Unpacking an explainable music recommendation system into patterns. Image adapted from [47]

A.1.5 Explainable Music Recommendation System (Figure 15). The proposed system aims to explore the design of explanations in a music recommender system in order to fit the user's preferences (selected songs, audio preferences) and personal characteristics (i.e., need for cognition, musical sophistication and openness). The system provides recommendations to the user based on a source song (i.e., a playlist song) and user preferences for audio features (danceability, energy, happiness, and popularity). User can search and add recommended tracks to the playlist, remove

existing track from the playlist, select a playlist song as a source song for recommendations, set their preferences through the audio features, and request and explore explanations. Explanations could be requested (and provided) both for a selected recommendation as well as for all songs at once.

We identify the following types of interaction: (a) model visualizes playlist and recommendations with control options – user adds (or removes) a song from the playlist/source list songs, as well as through the audio feature preferences, and (b) user can request further explanations for a given track or all songs – model provides visual and textual explanations to justify recommendations. In terms of the communicated information, the model and the user interact through showing and selecting (explainable) recommendations and audio feature preferences. In terms of implementation, the model utilizes user's actions (adding/removing songs, setting preferences, asking for XAI), in order to provide personalized recommendations and explanations. After certain interactions with the user, i.e., add/remove/search song or change audio preferences, the model updates its recommendations (updated feature vector). In terms of design aspects, the model provides different types of explanations in order to match the individual's preference and characteristics. Based on the outcomes from a set of user studies, the authors provide a set of design suggestions towards selecting appropriate explanation styles and levels of transparency based on user's personal characteristics. For example, users with low musical sophistication may prefer brief explanations that do not require domain knowledge. we can define the following actions (Table 14):

	Message	Action definition
F1	<model→user,**req-recommendations(R,UM)**, [[UM:audioPrefs,srcTrack;R:reccTracks]>	req-recommendations(R,UM)≡ request(R:[output.item],UM:input.fvector)) ← select(R),map(UM,R)
F2	<model→user,**show-recommendations(R,UM)**, [[UM:audioPrefs,srcTrack;R:reccTracks]>	show-recommendations(R,UM)≡ provide(R:[output.item],UM:input.fvector)) ← select(R),map(UM,R)
F3	<user→model,**modify-features(UM,M)**, [UM:audioPrefs;M:modifyPrefs,UIslider]>	modify-features(UM,M)≡ provide(M:input.fvector,UM:input.fvector) ←modify(UM,M)
F4	<model→user,**evaluate-recommendation(F,S)**, [F:addPlaylistTrack;S:selTrack]>	evaluate-recommendation(F,R)≡ provide(F:feedback.eval,S:output.item) ←select(S),map(S,F)
F5	<user→model,**modify-features(UM,M)**, [UM:srcTrack;M:newSrcTrack,click]>	modify-features(UM,M)≡ provide(M:input.fvector,UM:input.fvector) ←modify(UM,M)
F6	<model→user,**req-prediction_XAI(V,S,UM)**, [V:clickXAIBtn;S:selTrack;UM:audioPrefs,srcTrack]>	req-prediction_XAI(UM,S,V) ≡ request(V:feedback.XAI;[UM:input.fvector, S:output.item]) ← select(S), map(V,S,UM)
F7	<model→user,**show-prediction_XAI(V,S,UM)**, [V:openUserModel;S:selTrack;UM:audioPrefs,srcTrack]>	show-prediction_XAI(UM,S,V) ≡ provide(V:feedback.XAI;[UM:input.fvector, S:output.item]) ← select(S), map(V,S,UM)

Table 14. Messages and action definitions for the interactive sound annotation system interactions

We define the following patterns (Table 15): [F1] recommendations, [F2–F3] user-control-feedback, and [F4–F5] prediction-based_XAI. Based on the first pattern, the user is provided with a list of recommended tracks for the current preferences. The second pattern described the user's interactions with the recommendations and their preferences. The model updates its decisions based on the these user actions. The third pattern is part of an XAI-interaction where the system visualizes the preference model and the similarity to justify a given recommendation. The user makes the final decision about a recommended track (output) and the feature values (input).

pattern	actions	description
recommendations	`req-recommendations(UM,TR)`	user requests recommendations for preferences
	`show-recommendations(UM,TR)`	model visualizes recommendations for preferences
user-control-feedback	`modify-features(UM,M)`	user sets preferences and updates feature vector
	`evaluate-recommendation(S,PL)`	user adds or removes playlist song
prediction-based_XAI	`req-prediction_XAI(S,V,UM)`	user requests XAI for recommendations
	`show-prediction_XAI(S,V,UM)`	model visualizes user model

Table 15. Interaction patterns for the explainable music recommendation system

A.1.6 Transparent Meeting Scheduling Assistant (Figure 16). The proposed system uses an AI model to automatically detect meeting requests from free-text emails. The scheduling assistant provides the user with additional information (XAI), including an accuracy indicator component and a textual description for example-based explanations. The accuracy indicator visualizes the model's accuracy for the predictions in the form of a chart. The example-based explanations aim to enhance user's understanding about the underlying AI model and include a set of example sentences (inputs) and the model prediction (output) for each sentence, which can vary from "very unlikely" to "very likely" to describe the model's confidence, and depend on the model's sensitivity. The user is able to control model's sensitivity and see the updated results, through a UI slider which provides information about how sensitivity affects model's decisions.

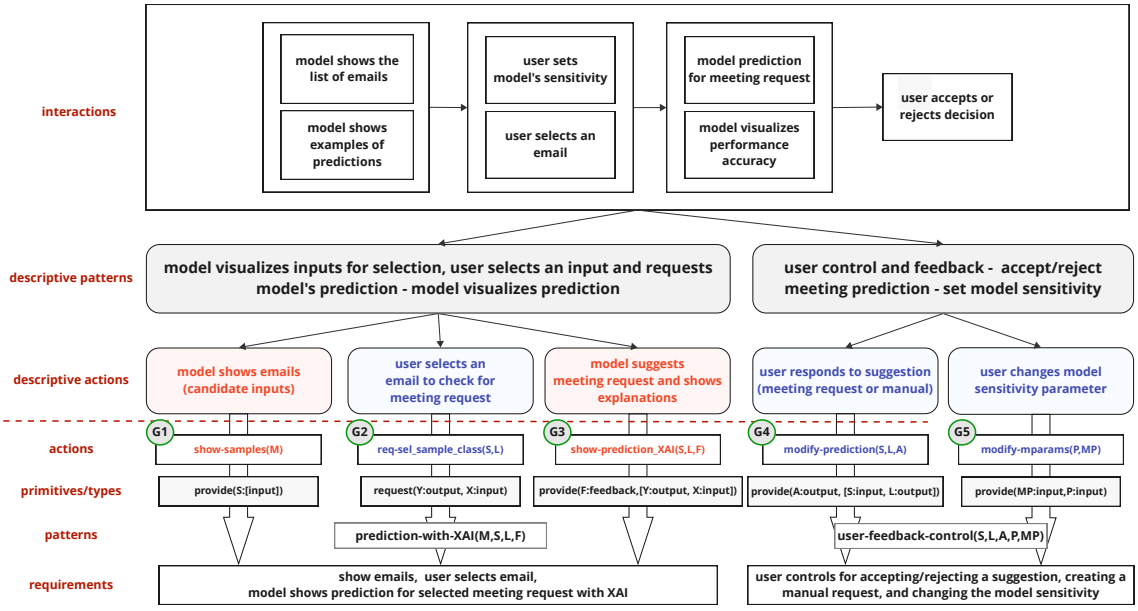

Fig. 16. Unpacking the meeting scheduling assistant [40] into patterns and primitives.

We can identify the following interactions: (a) model visualizes the inputs (e-mails) and user selects an item to check model's prediction - model provides prediction with XAI, (b) user provides feedback to the model through accepting or rejecting suggestions (predictions) and/or setting the sensitivity value through the slider. During these

interactions, user and model communicate messages through visualization and selection of model inputs, textual and visual explanations and control interfaces (slider), as well as accepting or rejecting model's decisions. IN order to enhance user's decision making, model provides explanations and transparency in terms of performance. Interactions with low-performance models (low confidence) can be effective for the user, if they provide an appropriate level of transparency and explainability. Such interaction can enhance user's decision making. In terms of implementation aspects, the different types of user feedback (e.g.,accepting/rejecting suggestions) can be used to update the model in an interactive way. Users can set the model sensitivity parameter value based on their observations of how it affects model's decisions. we define the following actions (Table 16):

	Message	Action definition
G1	<model→user,**show-samples(M)**, [M:emailList,freeText]>	show-samples(M)≡ provide(M:[input.raw_data])) ← select(M)
G2	<user→model, **req-sel_sample_class(S,L)**, [S:selectedEmail;L:L:isMeeting]>	req-sel_sample_class(S,L)≡ request(L:output.label,S:input.raw_data) ← select(S),map(S,L)
G3	<model→user, **show-prediction-XAI(S,L,F)**, [S:selectedEmail;L:L:isMeeting;F:XAIexamples]>	show-prediction-XAI(S,L,F)≡ provide(F:feedback.XAI;[S:input.raw_data,L:output.label]) ← map(S,L), map(F,S,L)
G4	<model→user,**modify-prediction(S,L,A)**, [S:selectedEmail; L:isMeeting;A:acceptBtn,createBtn]>	modify-prediction(S,L,A)≡ provide(A:output.label,[S:input.raw_data, L:output.label])← select(S),modify(L,A),map(S,A)
G5	<model→user,**modify-mparams(P,MP)**, [P:sensitivityValue; MP:modifiedValue,UIslider]>	modify-mparams(P,MP)≡ provide(MP:input.model_params,P:model_params)) ← modify(P,MP)

Table 16. Messages and action definitions for the meeting scheduling assistant

We identify two patterns: [G1-G3] prediction-with-XAI, and [G4-G5] user-feedback-control (Table ??). The first pattern describes an XAI-based interaction for input selection and prediction, where the model provides textual explanations of prediction examples to the user to enhance their understanding about the model's prediction. The second pattern describes a feedback control pattern; the user provides feedback to set the model sensitivity and interacts with the model decisions. Depending on the prediction, the user can agree with the model and accept a predicted request (true positive) or a predicted non-request (true negative). If the model does not predict a meeting request (false negative) the user can manually create a meeting request. If the model predicts a false meeting request (false positive), the user can ignore the predicted request.

pattern	actions	description
prediction-with-XAI	show-samples(M)	model visualizes all inputs (emails)
	req-sel_sample_class(S,L)	user selects an email and asks for suggestion
	show-prediction_XAI(S,L,F)	model visualizes prediction with XAI
user-feedback-control	modify-prediction(S,L,A)	user accepts or modifies prediction
	modify-mparams(P,MP)	user sets (a new) model sensitivity parameter

Table 17. Interaction patterns for the transparent meeting scheduling assistant

A.1.7 Explainable-driven Interactive Machine Learning for game outcome prediction (Figure 17). The proposed system integrates an explanation-driven interactive machine learning (XIML) mechanism to improve user's trust and satisfaction during the interaction with the system. The use case is the Tic-Tac-Toe game, where user and model make predictions about the winner of the game for a given state (game instance), in a turn-taking interaction. Both user and model can justify their predictions using rule-based explanations.

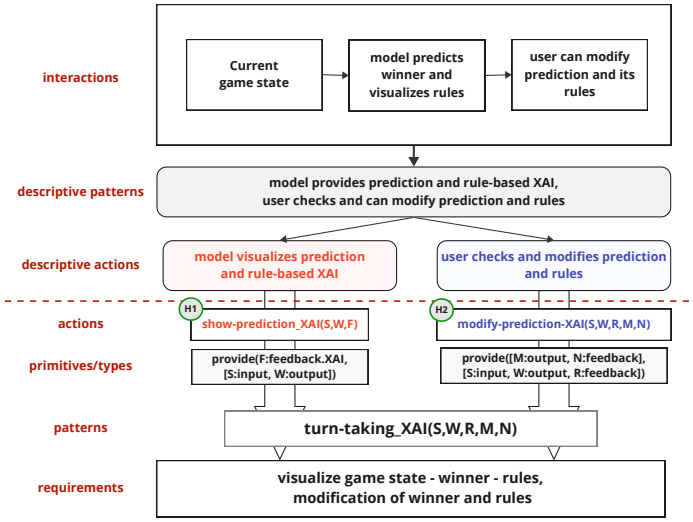

Fig. 17. Unpacking an interactive ML system for game rules into patterns and primitives. Image adapted from [34]

	Message	Action definition
G1	<model→user, **show-prediction_XAI(S,W,R)**, [S:gameState;W:predWinner;R:XAIrules]>	show-prediction-and-XAI(S,W,R)≡ provide(R:feedback.XAI;[S:input.raw_data,W:output.label]) ← map(S,L), map(F,S,L)
G2	<model→user,**modify-prediction-and-XAI(S,W,R,M,N)**, [S:gameState;W:predWinner;R:XAIrules; M:modifiedPred;N:modifiedRules]>	modify-prediction-and-XAI(S,W,R,M,N)≡ provide([M:output.label,N:feedback.XAI], [S:input.raw_data,W:output.label,R:feedback.XAI]) ← modify(W,M),modify(R,N),map(S,M,N)

Table 18. Messages and action definitions for the XAI-based interactive ML for game rules

During this turn-taking interaction, the model visualizes a game instance and its prediction for the winner. In order to justify its prediction, it visualizes the rule based on which the decision was made. The user can accept or modify both prediction and rules. The rules are a set of Boolean rules in disjunctive normal form (DNF). The authors conducted a user study to evaluate the effects of interactivity and visualization on user's trust and satisfaction. The outcomes of their analysis indicate that both aspects can have an effect on users' perception of control over the different types of visualization. The model visualizes its prediction and reasoning to enhance user's trust in model performance and does not update its prediction based on user 's input. However, similar interactions can take place in hybrid intelligence systems, where both users and models support their own decisions/predictions in order to augment each other's perception. Table 18 shows the action definitions of the pattern (Table 19). `turn-taking_XAI` a turn-taking pattern where both user and model can exchange the same information in an interactive manner.

pattern	actions	description
turn-taking_XAI	show-prediction_XAI(S,W,R)	model provides rule-based explanations for prediction
	modify_prediction_rules(S,W,MW,MR)	user accepts or modifies prediction and rules

Table 19. Interaction patterns for the explainable/interactive game outcome predictions