Model-based Multi-agent Reinforcement Learning for AI Assistants

Mustafa Mert Çelikok
Model-based Multi-agent Reinforcement Learning for AI Assistants

Mustafa Mert Çelikok

A doctoral thesis completed for the degree of Doctor of Science (Technology) to be defended, with the permission of the Aalto University School of Science, at a public examination held at the lecture hall Otakaari 1–Y405 of the school on 5 June 2023 at 11:00.

Aalto University
School of Science
Computer Science
Probabilistic Machine Learning
Supervising professor
Professor Samuel Kaski, Aalto University, Finland

Thesis advisor
Professor Samuel Kaski, Aalto University, Finland

Preliminary examiners
Doctor Goran Radanović, Max Planck Institute for Software Systems, Germany

Assistant Professor Hendrik Baier, Eindhoven University of Technology, The Netherlands

Opponent
Associate Professor Matthew E. Taylor, University of Alberta, Canada

Aalto University publication series
DOCTORAL THESES 51/2023

© 2023 Mustafa Mert Çelikok

ISSN 1799-4934 (printed)
ISSN 1799-4942 (pdf)

Unigrafia Oy
Helsinki 2023

Finland

Publication orders (printed book):
mustafa.mert.celikok@gmail.com
Abstract

Interaction of humans and AI systems is becoming ubiquitous. Specifically, recent advances in machine learning have allowed AI agents to interactively learn from humans how to perform their tasks. The main focus of this line of research has been to develop AI systems that eventually learn to automate tasks for humans, where the end goal is to remove the human from the loop, even though humans are involved during training. However, this perspective limits the applications of AI systems to cases where full automation is the desired outcome. In this thesis, we focus on settings where an AI agent and a human must collaborate to perform a task, and the end goal of the AI is not to replace human intelligence, but to augment it.

AI-assistance for humans involves at least two agents: an AI agent and a human. System designers have no control over the humans, and must develop learning agents that have the capabilities to assist and augment them. To do so, the AI agent must be able to infer the goals, bounds, constraints, and future behaviour of its human partner. In this thesis, we propose a model-based multi-agent reinforcement learning approach, where the AI agent infers a model of its human partner, and uses this model to behave in a way that is maximally helpful for the human.

In order to learn a mathematical model of the human from interaction, the AI agent first must have a model space. Since data scarcity is a key problem in human--AI collaboration, defining a model space that is expressive enough to capture human behaviour, yet constrained enough to allow sample-efficient inference is important. Determining the minimal and realistic set of prior assumptions on human behaviour in order to define such model spaces is an open problem. To address this problem, we bring in prior knowledge from cognitive science and behavioural economics, where various mathematical models of human decision-making have been developed. However, incorporating this prior knowledge in multi-agent reinforcement learning is not trivial. We demonstrate that, using the methods developed in this thesis, sufficient statistics of human behaviour can be drawn from these models, and incorporated into multi-agent reinforcement learning.

We demonstrate the effectiveness of our approach of incorporating models of human behaviour into multi-agent reinforcement learning in three types of tasks where: (I) The AI must learn the preferences of the human from their feedback to assist them, (II) The AI must teach the human conceptual knowledge to assist them, (III) The AI must infer the cognitive bounds and biases of the human to improve their decisions. In all tasks, our simulated empirical results show that the AI agent can learn to assist the human and improve the human--AI team’s performance. Our user study for the case (I) supports the simulated results. We present a theoretical result for case (III) which determines the limits of AI-assistance when the human user disagrees with the AI.

Keywords Reinforcement Learning, Multi-agent Learning, Human--AI Collaboration, Probabilistic Methods
Abstract

Interaction of humans and AI systems is becoming ubiquitous. Specifically, recent advances in machine learning have allowed AI agents to interactively learn from humans how to perform their tasks. The main focus of this line of research has been to develop AI systems that eventually learn to automate tasks for humans, where the end goal is to remove the human from the loop, even though humans are involved during training. However, this perspective limits the applications of AI systems to cases where full automation is the desired outcome. In this thesis, we focus on settings where an AI agent and a human must collaborate to perform a task, and the end goal of the AI is not to replace human intelligence, but to augment it.

AI-assistance for humans involves at least two agents: an AI agent and a human. System designers have no control over the humans, and must develop learning agents that have the capabilities to assist and augment them. To do so, the AI agent must be able to infer the goals, bounds, constraints, and future behaviour of its human partner. In this thesis, we propose a model-based multi-agent reinforcement learning approach, where the AI agent infers a model of its human partner, and uses this model to behave in a way that is maximally helpful for the human.

In order to learn a mathematical model of the human from interaction, the AI agent first must have a model space. Since data scarcity is a key problem in human–AI collaboration, defining a model space that is expressive enough to capture human behaviour, yet constrained enough to allow sample-efficient inference is important. Determining the minimal and realistic set of prior assumptions on human behaviour in order to define such model spaces is an open problem. To address this problem, we bring in prior knowledge from cognitive science and behavioural economics, where various mathematical models of human decision-making have been developed. However, incorporating this prior knowledge in multi-agent reinforcement learning is not trivial. We demonstrate that, using the methods developed in this thesis, sufficient statistics of human behaviour can be drawn from these models, and incorporated into multi-agent reinforcement learning.

We demonstrate the effectiveness of our approach of incorporating models of human behaviour into multi-agent reinforcement learning in three types of tasks where: (I) The AI must learn the preferences of the human from their feedback to assist them, (II) The AI must teach the human conceptual knowledge to assist them, (III) The AI must infer the cognitive bounds and biases of the human to improve their decisions. In all tasks, our simulated empirical results show that the AI agent can learn to assist the human and improve the human–AI team’s performance. Our user study for the case (I) supports the simulated results. We present a theoretical result for case (III) which determines the limits of AI-assistance when the human user disagrees with the AI.
Preface

I was privileged enough to be a member of the Probabilistic Machine Learning (PML) group in Aalto University for 6 years, from my Master’s to the end of my PhD. Throughout this time, I had the chance to work with exceptional people. I want to thank Dr. Tomi Peltola, who has provided me excellent guidance in terms of being a good researcher. Specifically, he taught me not to be disheartened when I discover that an idea of mine has already been published. I will carry his maxim of *they could not have solved all of it* with me forever. There is always future work, there is always more to do. I also want to thank Dr. Pierre-Alexandre Murena, similarly for showing me different ways to do research, and to follow your ideas through. A special thanks to my colleagues Dr. Pedram Daee, Sebastiaan de Peuter, Dr. Charles Gadd, and Zeinab Rezaei Yousefi for excellent discussions. Of course finally, a big thank you to my supervisor prof. Samuel Kaski, from whom I learned the most important and fundamental skills a researcher must possess. Thank you for teaching me how to think and write clearly, to identify important directions of research, and most important of all, to keep going. The opportunities you have provided me and the confidence you have placed are invaluable.

Thanks to my supervisor, prof. Kaski, I had the opportunity to be part of the European Laboratory for Learning and Intelligent System’s PhD programme. Thus, I visited prof. Frans A. Oliehoek’s group in Delft University of Technology during my PhD. This collaboration has proven most fruitful, and I have learned most of what I know in multi-agent learning from it. I thank prof. Oliehoek for hosting me, and treating me no differently from his own students. As part of my stay in Delft, I have had the chance to work with brilliant researchers such as Dr. Robert Loftin, Dr. Alexander Czechowski, Miguel Suau de Castro, Rolf Starre, and Jinke He.

I remember reading that a good scientist takes both life and science too seriously to separate them. Indeed, none of this work would have been possible without the endless support of my beloved partner, Mari-Liis Maikalo. Thank you for all your support, and for showing me that the growing pains are, in your words, a part of the creative process. Accepting
the struggle as part of the process makes all the difference. Finally, a late addition to the family, our Scottish terrier, Oliver... Apart from your emotional support, you taught me determination, by letting absolutely nothing get in between you and that one spot you have to sniff.

The articles comprising this thesis were financially supported by the Academy of Finland (Flagship programme: Finnish Center for Artificial Intelligence, FCAI; grants 319264, 313195, 305780, 292334, 328400, 28400), the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement No. 758824 —INFLUENCE), the UKRI Turing AI World-Leading Researcher Fellowship EP/W002973/1, ELISE travel grant (GA no 951847), KAUTE Foundation, and the Aalto Science-IT Project. My doctoral work is funded by the Finnish Science Foundation for Technology and Economics KAUTE and the Aalto University Department of Computer Science Competitive Co-funding Scheme. I acknowledge the computational resources provided by the Aalto Science-IT Project.

Helsinki, March 31, 2023,

Mustafa Mert Çelikok
### 1. Introduction

1.1 Motivation .................................................. 15
1.2 Research Questions and Contributions .................. 16
1.3 Organization of the Thesis ................................. 19

### 2. Model-based Approaches for Human–AI Collaboration

2.1 Human–AI Collaboration .................................... 21
2.2 Single-Agent Reinforcement Learning ................... 22
  2.2.1 Markov Decision Problems ............................. 23
2.3 Model-based Reinforcement Learning .................... 27
2.4 Partially-observable Markov Decision Process ......... 31
2.5 Multi-agent Reinforcement Learning and Human-AI Collaboration .......................... 34
  2.5.1 Partially-observable Stochastic Games ............... 35
  2.5.2 Bayesian Best-response Models ....................... 36
2.6 Conclusion .................................................. 38

### 3. Contributions

3.1 A Machine Teaching Framework for Human-AI Collaboration (Publications I and II) .................. 41
  3.1.1 Machine Teaching ...................................... 41
| 3.1.2 | Learning from Strategically-Steering Humans (Publication I) | 42 |
| 3.1.3 | Teaching Humans for Effective AI Assistance (Publication II) | 45 |
| 3.2 | A Bayesian Multi-agent Reinforcement Learning Framework for Human-AI Collaboration (Publications III) | 49 |
| 3.2.1 | Improving Human Intelligence with Centaurs | 49 |
| 3.2.2 | Theoretical Bounds of Improving Human Intelligence with Centaurs | 53 |
| 3.3 | A Call-to-arms for Advancing User Modelling Research in Human-centred Machine Learning (Publication IV) | 55 |

4. Concluding Remarks 57

References 59

Errata 67

Publications 69
This thesis consists of an overview and of the following publications which are referred to in the text by their Roman numerals.


Author’s Contribution

Publication I: “Machine Teaching of Active Sequential Learners”

The lead authors of the article were (A) Tomi Peltola and (DC) Mustafa Mert Çelikok. DC and A contributed to all parts of the paper in varying degrees. DC contributed more to developing the model, whereas A contributed more to inference. DC and A performed and analysed the simulated experiments. Pedram Daee performed the user experiments with help from A and DC. Samuel Kaski was the supervising professor and provided guidance.

Publication II: “Teaching to Learn: Sequential Teaching of Learners with Internal States”

The lead author of the article was (DC) Mustafa Mert Çelikok. DC contributed to all parts of the paper: developed the novel model, implemented its inference, performed and analyzed all of the experiments, and assisted (A) Pierre-Alexandre Murena in deriving the theoretical results. A derived theoretical results, and provided guidance to DC. Samuel Kaski was the supervising professor and provided guidance.

Publication III: “Best-Response Bayesian Reinforcement Learning with Bayes-adaptive POMDPs for Centaurs”

The lead author of the article was (DC) Mustafa Mert Çelikok. DC contributed to all parts of the paper: developed and implemented the novel model, performed and analyzed all of the experiments, and derived the theoretical results. Samuel Kaski and Frans A. Oliehoek were the supervising professors and provided guidance.
Publication IV: “Modelling Needs User Modelling”

Mustafa Mert Çelikok and Pierre-Alexandre Murena have contributed equally to the writing of this paper, under the guidance of the supervising professor Samuel Kaski.
List of Figures

3.1 The Rock Sample Environment from [1]. . . . . . . . . . . . 53
Abbreviations

AI  Artificial Intelligence

MDP  Markov Decision Process

POMDP  Partially-Observable Markov Decision Process

RL  Reinforcement Learning

ML  Machine Learning

POSG  Partially-Observable Stochastic Game

ToAI  Theory of AI’s Mind

ToM  Theory of Mind
1. Introduction

1.1 Motivation

Recently, the field of artificial intelligence has seen an incredibly rapid pace of development, spearheaded by advances in machine learning. Most importantly, there has been a strong focus on building autonomous systems that can perform tasks without human assistance. We will call this case full automation. For instance, the main objective of developing self-driving cars is to automate the driving process in a way that cars can drive themselves without any human input. Indeed, certain mechanical tasks are suitable for full automation, however intelligent agents such as humans also collaborate with each other in order to perform tasks together. This thesis focuses on the case where a human and an AI agent must perform a task together collaboratively, where full automation is not the end goal. Here, the AI agent’s objective is not to replace the human, but to work together with them. There are research questions and problems uniquely associated with developing AI agents that can engage in such collaboration with humans. This thesis aims to address some of these questions, which are enumerated in section 1.2.

Humans and AI systems have different strengths and weaknesses. It is safe to state that modern AI systems have access to greater raw computational power, however human intelligence is undoubtedly more complex and capable of abstract thought. On the other hand, humans have limited cognitive resources, which may lead to suboptimal behaviour. In addition, the cognitive shortcuts we have learned throughout our evolution may in many cases hinder our decision-making. This implies that the AI and humans have complementary strengths, and if we can build AI systems that can collaborate with humans, we can augment human intelligence. The case where the human and artificial intelligence are combined to augment human intelligence has recently been proposed as hybrid intelligence [2].

A main argument of this thesis is that if an AI is to augment a human's
intelligence, the AI must recognize the existence of its human partner by modeling them. This is indeed the case when humans are collaborating with other humans, and the phenomenon of modeling the minds of others is called the *theory of mind* in cognitive science [3, 4, 5, 6].

To give an intuitive example, imagine an AI agent assisting a person in choosing a specific item from a set of items. It might be the case that the most discriminating feature of the right item is its colour, and thus the optimal thing to do for the AI is to let the person know the item’s colour. However, if the person in question is colour-blind, item colour may no longer be the most discriminating feature. An AI agent equipped with a model of its human partner may infer from their interaction that the human is colour-blind, and change strategy for future such tasks. In any case where we expect an AI to collaborate with a human in order to augment them, the AI’s strategy will depend on the modelling assumptions and inferences it makes about the human partner. This thesis is motivated by the need for a rigorous mathematical framework that can represent such human-AI collaboration settings.

### 1.2 Research Questions and Contributions

This thesis proposes a mathematical framework for human-AI collaboration scenarios where the goal of an AI agent is to augment the human intelligence instead of replacing it. The unifying theme of the thesis is that to augment its human partner, an AI agent must be able to infer a model of its partner. In order to endow an agent with such capabilities, system designers must make certain design choices that are crucial for success. For instance, the first design decision to make is the form of interaction between the AI and the human (called the *interaction protocol* henceforth), which involves determining the roles and privileges of each agent. In this thesis, each one of these design choices are represented by motivating questions (MQ). The motivating questions are open problems, and we contribute to their solutions by formulating and investigating specific research questions (RQ) for each MQ.

**MQ1**  
*Can we develop practically plausible interaction protocols between an AI and a human that enable better collaboration between them, by allowing the AI to infer a model of the human and augment human intelligence?*  
Human intelligence involves acquiring and retaining knowledge, reasoning, planning, and goal-directed behaviour. The question is then what type of interaction protocols will allow the AI to augment which aspects of the human intelligence. We contribute to answering this question by developing and mathematically formulating two different interaction protocols for three different cases of human–AI collaboration. As a general rule, every interaction protocol induces a particular structure on what can
be assumed and inferred about the human partner.

**RQ1.1** - *Can a teacher–learner interaction protocol improve human–AI collaboration, and how can such a protocol be modelled computationally?* In publications I and II, the interaction between the human partner and the AI agent is modelled as a teacher–learner relationship. Specifically, in publication I, the human partner is modelled as the teacher, whereas in publication II, they are modelled as the learner. In both cases, the interaction is modelled under the framework of *machine teaching*, which will be discussed further in section 3.1.1. This interaction protocol models collaboration settings where the AI must either learn from a human teacher or must teach the human, in order to be able to assist the human.

**RQ1.2** - *Can an AI assistant learn to augment a human’s intelligence, when the human has the authority to override its actions?* Publication III proposes a supervisor–agent protocol where the human has the authority to override any action of the AI. This protocol is a superset of the teacher–learner relationship, which becomes a special case. The supervisor–agent protocol is able to model a much richer class of settings, where the human for instance may be monitoring the behaviour of an autonomous agent such as a self-driving car.

**MQ2** - *What does the AI need to assume of the human, so that it can infer a model of them?* The assumptions that are needed and plausible depend upon the task and the interaction protocol. We contribute to this question in three cases, where publications I, II, and III explore the usefulness of different advanced models of the human partner in their respective interaction protocols and tasks.

**RQ2.1** - *What are the needed and plausible assumptions about the human in the case of the teacher–learner interaction protocol, where the human may be either the teacher or the learner?* In publication I, the AI’s task is to infer the preferences of the human in order to help them achieve their goal. Here, the AI models the human partner as a mixture of two behavioural models: a fixed probability distribution based on the human preferences and a decision-making agent who plans into the future in order to actively steer the AI. In the latter case, the human is modelled as they are trying to *teach* the AI their desired behaviour strategically. The AI uses this mixture model as its likelihood function to interpret the human feedback it receives, in order to infer the preferences of the human partner. Publication II swaps the roles of the human and the AI, where the AI models the human partner as a learner, and attempts to teach them in order to assist them more effectively. More specifically, the AI agent recognizes that its assistance may be refused even though helpful, if the human partner lacks in conceptual knowledge. It maintains a model of how this knowledge affects the partner’s decision-making, infers
Introduction

their knowledge level from interaction, and attempts to teach the partner necessary concepts to assist them further.

**RQ2.2** - What are the needed and plausible assumptions about the human in the case of the supervisor–agent interaction protocol, where the human has the authority to override the AI's actions? In publication III, the human is modelled as a decision-maker that follows a policy conditioned on their internal state, which they maintain using their subjective model of the world. The human's subjective model may differ from the AI agent's, which leads to disagreements between the human and the AI. Within the three publications, this model is the most general one, which can mathematically represent the models in publication I and II.

**MQ3** - What are the theoretical limitations of Bayesian multi-agent reinforcement learning for human–AI collaboration? An important question is the theoretical limits of what can be learned through Bayesian multi-agent reinforcement learning for human–AI collaboration. Motivated by this open question, we formulated and investigated the following research question.

**RQ3.1** - How does the underlying mathematical structure of the task affects the convergence rate of the beliefs of two agents when these agents model the task differently, and how does the difference in beliefs affect collaboration in the supervisor–agent setting? Publication III identifies a problem called the belief alignment problem, where the human and the AI may have different beliefs about the state of the world, and these beliefs may never come closer if the two agents model the world differently. It also demonstrates that the belief alignment problem can lead to failure to collaborate between the human and the AI. Then, the article presents a theoretical result on the contraction rate of the human and the AI's beliefs over time. This is a structural result which identifies desirable properties for the underlying dynamics and observation models in order to enable belief contraction.

Throughout our work in publications I, II, and III, we have identified important challenges and dimensions to consider when building advanced models of human partners. In order to facilitate further research, we have written the publication IV, which presents a unified perspective on modelling human users of AI systems. It presents our more general findings in an organized manner, by specifying three important dimensions which must be considered when building advanced models of human partners: (1) Humans have goals, some of which may be tacit, (2) Humans are bounded-rational, and (3) Humans may model the AI systems they use and engage in recursive reasoning. In addition, it identifies four open problems and challenges: (1) The scarcity of human-AI interaction data, (2) The non-stationary nature of human behaviour, (3) Computational complexity of advanced models of users, and (4) The learnability issues pertaining to
models of human behaviour. Publication IV aims to contribute to the community with a research direction and vision in modelling human users for human–AI collaboration, and serves as a starting point for future research.

1.3 Organization of the Thesis

This thesis is prepared according to the article-based thesis guidelines of Aalto University. Therefore, it consists of a set of publications on a related set of problems, and a summary of the findings. The chapter 2 provides a concise background on single and multi-agent reinforcement learning, and human-AI collaboration. Then, the chapter 3 provides summaries of the contributions of the comprising articles (i.e. the publications included in this thesis). Finally, chapter 4 provides the concluding remarks. The articles comprising this thesis are appended to the end.
2. Model-based Approaches for Human–AI Collaboration

In this chapter, we will give a brief background on the concepts that are necessary to understand the contributions of the comprising articles. First, we will give a brief description of what do we mean by human–AI collaboration in this thesis, in section 2.1. Then, section 2.2 provides an overview of single agent reinforcement learning, and the section 2.5 describes the main multi-agent reinforcement learning models used in comprising articles, from the perspective of human-AI collaboration. In all the comprising articles, the human-AI collaboration is a multi-agent problem, however, in certain cases the problem can be reduced into an equivalent single agent form with special structure.

2.1 Human–AI Collaboration

The concept of human–AI collaboration is possibly as old as the concept of AI. It describes the general case of an AI system and human(s) working together towards a goal. Thus, in its broadest definition, it captures all of machine learning, since in the end even the simplest machine learning system is designed and trained by humans, and eventually serves a purpose for its designers.

In this thesis, we consider a more specific case of human–AI collaboration where an AI agent and a human (or humans, but for simplicity we will assume a single human) must together perform a sequential decision-making task with delayed reward. This means mathematically, we deal with either fully cooperative or general-sum sequential games [7], where different agents must work together to find good solutions to a sequential decision-making task. For example, a human and an AI may need to coordinate in order to help evacuate people from a burning building. Solving this problem requires making a sequence of decisions, while making sure your decisions work well together with the decisions of the other agent. What distinguishes human–AI collaboration from the general case of collaboration in sequential games is the fact that one of the agents is a
human while the other is an AI designed by us. This difference imposes certain restrictions on how we can learn, what we can learn, and what assumptions can we make about the other agents. For instance, learning in simulation is not enough any more, since in the end we have to interact with real humans in real tasks. This brings in sample complexity and safety constraints. We also cannot take certain properties of the other agent as granted, such as perfect rationality.

In this thesis, we take the model-based approach to human–AI collaboration, where both the AI and the human are considered as decision-making agents, and the AI infers a model of its human partner. This is in contrast to other possible approaches, such as a purely human–computer interaction approach which might consider the AI as a passive tool [8], where the human is the decision-maker. More closely related to our approach, a model-free reinforcement learning approach would avoid modelling the human decision-making, and consider the human as simply a part of the physical environment the AI operates in, which is not modelled explicitly. The benefits of learning models of humans compared to model-free approaches for human–AI collaboration have been recently investigated and demonstrated with simple models [9]. Our work goes one step further into inferring more advanced models of human decision-makers to improve sample efficiency of learning and performance of collaboration.

2.2 Single-Agent Reinforcement Learning

In general, reinforcement learning is learning how to act in an environment in order to achieve a certain goal. There may be only one or multiple agents in an environment, each with their own goals, which may or may not be the same. Therefore, multi-agent reinforcement learning is a particularly suitable method for human-AI collaboration. In certain cases, the multi-agent setting can be reduced to single agent in a principled way. However, this does not mean starting from the single agent case is the right approach. Even if a setting can be reduced to single agent, the reduction from multiagent to single agent provides us with structural information which can be leveraged for sample and computational efficiency.

In this section, we will go over the main concepts of reinforcement learning, such as its models and solution methods. The field of single and multi-agent reinforcement learning lies at the intersection of machine learning, control theory, game theory, dynamical systems theory, and operations research. Therefore, a full treatment takes more than a single book, and we only focus on models and algorithms that are relevant to the comprising articles of this thesis.
2.2.1 Markov Decision Problems

In reinforcement learning, when there is a single agent acting in the environment, the decision-making task is often modelled as a Markov decision problem. We will first start by defining Markov decision processes (MDP), and then Markov decision problems. Throughout this thesis, we will always assume discrete-time models.

**Markov Decision Process (MDP).** MDPs are sequential decision processes, where decision epochs (also called time-steps) are indexed by time \( t \in \mathbb{N} \). Mathematically, an MDP is defined as the tuple

\[
M = (\mathcal{S}, \mathcal{A}, p_t(\cdot | s, a), r_t(s, a), H),
\]

(2.1)

where the set of states \( \mathcal{S} \) consists of different states the environment may be in, and the set of actions \( \mathcal{A} \) consists of all the actions an agent can take in the environment [10]. It may be that different states \( s \in \mathcal{S} \) have different possible actions, in which case \( \mathcal{A} = \bigcup_s \mathcal{A}_s \) where \( \mathcal{A}_s \) denotes the set of actions possible at the state \( s \). The \( \mathcal{S} \) and \( \mathcal{A} \) can be finite, countably infinite, or non-empty Borel subsets of complete, separable metric spaces. In this thesis, unless stated otherwise, we will focus on finite sets. The \( p_t(\cdot | s, a) \) represents the transition probabilities, where \( p_t(s' | s, a) \) denotes the probability of the environment transitioning to state \( s' \) from \( s \) when the action \( a \) is taken, at time-step \( t \). The reward function \( r_t : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R} \) determines the reward an agent receives when at \( s \) and time-step \( t \), they take action \( a \). We will always assume that the reward \( r_t \) is bounded everywhere. Finally, the horizon \( H \in (1, \infty] \) determines how many time-steps are there in the decision-making process, before it terminates. The MDP is called *infinite horizon* when \( H = \infty \), *finite horizon* when \( H < \infty \) is fixed and known, and *indefinite horizon* when \( H < \infty \) is a random variable.

The Markov epithet here is due to the fact that both rewards and transition probabilities depend only on the current state \( s \). An alternative way to look at this is to say that state \( s \) is a sufficient statistic for an agent to make decisions within this MDP. Historically, the elements of an MDP were designed carefully so that \( \mathcal{S} \) represents all the relevant information for the task at hand. Recently, machine learning approaches have begun to emerge which try to learn state representations or abstractions from raw observations.

An MDP is in the end a decision-making model, where an agent must decide which actions to take. Such decisions are formalized by decision rules. There are four classes of decision rules: Markovian Deterministic (MD), Markovian Stochastic (MS), History-dependent Deterministic (HD), and History-dependent Stochastic (HS). Markovian decision rules are functions that have \( \mathcal{S} \) as their domain, whereas history-dependent decision rules have the set of all possible histories \( \mathcal{H} \). This set is defined as a union
set of fixed-length histories $\mathcal{H} = \cup_t \mathcal{H}_t$, where $\mathcal{H}_t$ is the set of all possible t-step histories $(s_0, a_0, ..., s_t)$. Deterministic decision rules map directly to the set of actions, whereas stochastic ones map to a probability distribution over $\mathcal{A}$. Thus, for instance, an MD decision rule is denoted as $d_t : \mathcal{S} \rightarrow \mathcal{A}$ whereas an HS one is $d_t : \mathcal{H}_t \rightarrow \Delta(\mathcal{A})$ with $\Delta(\mathcal{A})$ denoting the set of probability distributions over $\mathcal{A}$. We let $D_i^t \mid i \in \{\text{MD, MS, HD, HS}\}$ denote the set of all class-$i$ decision rules for time-step $t$. A policy $\pi$ is essentially the collection of the decision rules an agent has for each time-step, throughout the entire horizon $H$. For instance, the set of policies that consist of MD decision rules is denoted by $\Pi^{\text{MD}} = D_0^{\text{MD}} \times D_1^{\text{MD}} \times ... \times D_H^{\text{MD}}$. Therefore, the policy is all that is needed to simulate an agent's behaviour in an MDP.

Apart from the class of their decision rules, policies are also distinguished by their time-dependence. A stationary policy consists of decision rules that do not depend on time, thus $\pi = (d, d, ...) \triangleq d^\infty$. The $\Pi^{\text{SS}}$ and $\Pi^{\text{SD}}$ denote the sets of stationary-stochastic and stationary-deterministic policies, respectively. Note that the sets of stationary policies are subsets of the sets of Markovian policies, and the sets of Markovian policies are subsets of the sets of history-dependent policies [11].

**Markov Decision Problem.** Given a class of policies $\Pi^i$ and an MDP $M$, how can we choose an optimal policy $\pi^* \in \Pi^i$ for $M$? In order to answer this question, we must define what optimal means. We have intentionally avoided answering this question in the definition of an MDP, because in this thesis we prefer the original definitions of [11] and require an explicit optimality criterion $\text{OPT}$ to be provided next to an MDP. Given an MDP $M$ and a policy class $\Pi^i$, an optimality criterion essentially provides us with a preference ordering over policies in $\Pi^i$. When an MDP is combined with an optimality criterion, it becomes a Markov decision problem. Here, we will describe the most popular optimality criteria for finite and infinite horizon MDPs. This list is not exhaustive, and the interested reader can find further pointers to more advanced criteria in [11, 12, 13].

The easiest way to define $\text{OPT}$ is through functions that accumulate reward over time. For instance, the expected total reward criterion can be defined via the function $v_H^\pi(s) \triangleq \mathbb{E}_M^{\pi} \left[ \sum_{t=0}^H r_t(s_t, a_t) \mid s_0 = s \right]$, where the expectation is taken with respect to policy $\pi$ and the transition probabilities of the MDP $M$. This function induces an ordering over policies such that if $v_H^{\pi_1}(s) \geq v_H^{\pi_2}(s), \forall s \in \mathcal{S}$ and $v_H^{\pi_1}(s) > v_H^{\pi_2}(s), \exists s \in \mathcal{S}$, then $\pi_1$ is preferred over $\pi_2$ with respect to this criterion. The definition of an optimal policy depends on whether $H = \infty$ or not. If $H \neq \infty$, then we can define an optimal policy as $\pi^* \in \arg \max_{\pi \in \Pi^i} v_H^\pi$, with respect to the $\text{OPT}$, the $M$, and the policy class $\Pi^i$. An important question is which policy class to choose, so that there is no better policy in any other classes. Unfortunately, for finite-horizon MDPs we cannot guarantee that the best policy is stationary. However, we can always guarantee that there is no better policy than the deterministic and Markovian optimal. Therefore, the theoretically
appropriate class to choose for finite-horizon MDPs is \( \Pi^{MD} \). In practice though, given finite computational power and highly complex (i.e. large state-action sets) problems, certain reinforcement learning algorithms may find better stochastic policies faster than deterministic ones. In this thesis, unless stated otherwise, we focus on deterministic policies.

Another important optimality criterion for finite-horizon MDPs is the **expected total discounted reward.** This criterion is defined as

\[
v_H^\pi(s) = \mathbb{E}_M^\pi[\sum_{t=0}^H \lambda(t) r_t(s_t, a_t) | s_0 = s]
\]

where \( \lambda(t) \to (0, 1) \) is a discount function. Readers who are familiar with discounting for infinite-horizon problems may find it unnecessary to discount rewards in finite-horizon ones. However, discounting has other uses than providing mathematical convergence for infinite-horizon sums. Discounting is crucial for understanding the publication III, thus we will go into further details of its use cases in section 3.2. In summary, the discount factor models the probability of an agent surviving to the next time step, and there is a close connection between the mathematical form of \( \lambda(t) \) and modelling assumptions about this probability [14].

We can define the expected total reward for infinite-horizon problems by taking the limit

\[
v_H^\pi(s) = \lim_{t \to \infty} \mathbb{E}_M^\pi[\sum_{t=0}^H r_t(s_t, a_t) | s_0 = s]
\]

However, we cannot guarantee that this limit exists, and even if it does, it may be \( \infty \) or \( -\infty \). For instance, if \( r_t(s_t, a_t) \) is positive for all \( (s_t, a_t) \), then the limit will be \( \infty \) regardless of the policy, which means this optimality criterion will not be able to distinguish policies. In turn, for instance, the limit may not exist if the sum \( \sum_{t=0}^H r_t(s_t, a_t) \) is periodic with respect to \( H \). One way to ensure that the infinite sum of rewards converge is to introduce discounting. Therefore, we define the expected total discounted reward for infinite horizon problems as

\[
v_H^\pi(s) = \lim_{t \to \infty} \mathbb{E}_M^\pi[\sum_{t=0}^H \lambda(t) r_t(s_t, a_t) | s_0 = s]
\]

where \( \lambda(t) \to (0, 1) \) is a discount function. Keep in mind that in order for this limit to converge, the one-sided limit \( \lim_{t \to \infty} \lambda(t) \) must exist. The most commonly used discount function is called the **exponential discount function**

\[\lambda(t) = \lambda^t\]

for a constant \( \lambda \in (0, 1) \). Clearly, \( \lim_{t \to \infty} \lambda^t = \frac{1}{1-\lambda} \), which means the infinite sum of rewards has the upper-bound \( \sum_{t=0}^\infty \lambda^t r_t(s_t, a_t) \leq \frac{R_{\text{max}}}{1-\lambda} \) where \( R_{\text{max}} \) is the maximum attainable reward at any \( (s_t, a_t) \). We will simply state that the expected total discounted reward is well-defined for any infinite-horizon MDP. A more detailed treatment of this statement’s proof and other optimality criteria such as average expected total reward can be found in [15].

**Value Functions and Bellman Optimality Equation.** In a Markov decision problem, the optimality criterion provides us a natural way to evaluate a policy. In general, evaluating a policy \( \pi \) means calculating the \( v_H^\pi(s) \) for every possible state \( s \in \mathcal{S} \), where we overload the notation \( v_H^\pi(s) \) to include both finite and infinite horizon criteria, and also allow \( \lambda = 1 \) to represent the undiscounted case. When it comes to the optimality criteria we have
described, there is a recursive way to write the evaluation equations. For $\Pi^{HR}$ (and thus also for Markovian or stationary policies since $\Pi^{HR}$ is their superset), we can define a function $V_{t,\lambda}^\pi(h_t) = \mathbb{E}_{a_t \sim d(h_t)}[r_t(s_t, a_t) + \lambda \sum_{s' \in S} p_t(s' | s_t, a_t) V_{t+1,\lambda}^\pi(h_{t+1} a_t s')]$ where $h_t a_t s' \in H_{t+1}$. It is easy to see that if we unroll the recursion of $V_{0,\lambda}(h_0)$, this gives $v_{H,\lambda}(h_0)$. The functions $V_{t,\lambda}^\pi$ are called value functions (or state value functions when $\pi$ is Markovian) in reinforcement learning literature. In essence, $V_{t,\lambda}^\pi(h_t)$ provides us the expected sum of rewards we get, if we follow policy $\pi$ from $t$ onwards, starting at $h_t$. This function can be computed thanks to the recursive formula, using dynamic programming. Let $V_{t,\lambda}^\pi(h_t) = \max_{\pi \in \Pi^{HR}} V_{t,\lambda}^\pi(h_t)$, then we call $V_{t,\lambda}^\pi(h_t)$ the optimal value function, since the policies that attain this function when evaluated are optimal policies. An important property of $V_{t,\lambda}^\pi(h_t)$ is given by the equation

$$V_{t,\lambda}^\pi(h_t) = \max_{a_t \in A_t} \{ r_t(s_t, a_t) + \lambda \sum_{s' \in S} p_t(s' | s_t, a_t) V_{t+1,\lambda}^\pi(h_{t+1} a_t s') \}. \quad (2.2)$$

The equation 2.2 is sometimes called the Bellman optimality equation.

If we were given the optimal value function $V_{t,\lambda}^\pi$, we would still need a description of the MDP in order to figure out the optimal policy. This is because in order to recover the optimal policy from $V^\pi$, we need to solve $\arg \max_{a_t \in A_t} \{ r_t(s_t, a_t) + \lambda \sum_{s' \in S} p_t(s' | s_t, a_t) V_{t+1,\lambda}^\pi(h_{t+1} a_t s') \}$, which involves computing an expectation over $p_t$. We can resolve this issue by defining a new function,

$$Q_{t,\lambda}^\pi(h_t, a) = r_t(s_t, a) + \lambda \sum_{s' \in S} p_t(s' | s_t, a_t) V_{t+1,\lambda}^\pi(h_{t+1} a_t s'). \quad (2.3)$$

It is straightforward to show that $\max_{a_t \in A_t} Q_{t,\lambda}^\pi(h_t, a) = V_{t,\lambda}^\pi(h_t)$. This function is called the optimal Q-function where $Q_{t,\lambda}^\pi(h_t, a)$ essentially answers the question of what would our expected sum of rewards be, if we were to take action $a$ at $h_t$, but then follow the optimal policy onwards. Now, if we are given the $Q_{t,\lambda}^\pi(h_t, a)$, we do not need anything else, since we can simply perform $\arg \max_{a_t \in A_t} Q_{t,\lambda}^\pi(h_t, a)$ to act in the environment. Just like the value function, the optimal Q-function also has a policy-based one where $Q_{t,\lambda}^\pi(h_t, a) = r_t(s_t, a) + \lambda \sum_{s' \in S} p_t(s' | s_t, a_t) V_{t+1,\lambda}^\pi(h_{t+1} a_t s')$. In this case, the function captures the expected sum of rewards of taking action $a$ at $h_t$, then following the policy $\pi$ onwards.

In this thesis, when the MDP is defined as infinite-horizon, we restrict ourselves to the set of stationary deterministic policies $\Pi^{SD}$ unless stated otherwise. Similarly, for finite-horizon settings, we restrict ourselves to the set of Markovian deterministic policies $\Pi^{MD}$. This is justified by the fact that for optimality criteria considered here, there always exists an optimal within these classes for their corresponding settings [15, Chapter 6]. We will also assume that all MDPs have stationary transition distributions and rewards, thus drop the time subscript to have $p(. | s, a)$ and $r(s, a)$.

26
The approaches to reinforcement learning can be divided into two main categories: (I) Model-based reinforcement learning and (II) Model-free reinforcement learning. For machine learning researchers who are unfamiliar with reinforcement learning, the term model-free might sound odd. This does not imply that there are no models, but that the method does not use an MDP (i.e., a model of the decision-making task) explicitly in its computation. In this thesis, we focus on model-based RL.

2.3 Model-based Reinforcement Learning

We are facing an unknown environment, where we need to make decisions. We know the state and action sets $\mathcal{S}, \mathcal{A}$, and we are given an optimality criterion $OPT$. Unfortunately, we do not know $p(. | s, a)$, and we may also not know $r(s, a)$ beforehand. How can we learn a good policy by interacting with the environment? Perhaps the most straightforward approach would be to try to learn $p(. | s, a)$ and $r(s, a)$ by collecting data from the environment. Then, once we are confident about our estimates $\hat{p}$ and $\hat{r}$, we can pretend that this is the true model of the environment, and compute the optimal policy. This approach is often named model-based, since it involves learning an explicit MDP model of the environment. Specifically, the model-based RL consists of two stages: learning a good MDP model of the environment, and then solving this MDP. These stages might be strictly separated, or interleaved. For a detailed and up-to-date account of model-based reinforcement learning, we refer the readers to the excellent survey of [16].

In general, model-based reinforcement learning methods are more sample-efficient than their model-free competitors. This makes them a natural fit for settings where interaction with the environment is limited or expensive, such as in human-AI collaboration. However, this is not to say they are perfect. Important issues such as model bias can make model-based reinforcement learning undesirable.

Sometimes, the line between model-free and model-based is blurred. For instance, deep Q-learning in itself is a model-free deep reinforcement learning method [17]. However, a variant of this algorithm uses experience replay, where the algorithm stores a buffer of data from the environment and uses it to perform optimization steps without directly interacting with the environment. This data can essentially be seen as an empirical model of the $p$ and $r$. Another method that explicitly aims to combine the two approaches is Dyna [18]. In addition to these cases, certain algorithms make use of sampling models, which are models from which an algorithm can generate samples of transitions $(s_t, a_t, r_t, s_{t+1})$. These can be simulators, which allow the agent to learn without acting in the real-world. Whether such algorithms be considered model-based or free is not strictly settled.
Bayesian Model-based RL and Bayes-adaptive MDP. Bayesian approaches to model-based RL start by placing a prior distribution over transition probabilities \( p(. \mid s,a) \), and in some cases over \( r(s,a) \) as well. For ease of exposure, we will assume the reward function is known. For finite state and action sets, \( p(. \mid s,a) \) is a categorical probability distribution, and the dynamics of such an MDP can be expressed by \( |\mathcal{S}| \times |\mathcal{A}| \) categorical distributions. This distribution is a generalization of the Bernoulli distribution, and each \( p(. \mid s,a) \) is parameterized by a \( |\mathcal{A}| \)-dimensional vector of probabilities. In Bayesian model-based RL, we place priors over these probability vectors. Fortunately, the categorical distribution has a conjugate prior, the Dirichlet distribution.

The Dirichlet distribution is the multivariate generalization of the Beta distribution, and has the following probability density function,

\[
\frac{\Gamma\left(\sum_{i=1}^{K} a_i\right)}{\prod_{i=1}^{K} \Gamma(a_i)} \prod_{i=1}^{K} x_i^{a_i-1},
\]

where \( K \geq 2 \) is the number of categories and \( \alpha = (\alpha_1, \ldots, \alpha_K) \) are the non-zero concentration parameters. The support of this probability density function is the \( K \)-dimensional probability simplex, thus samples from a Dirichlet distribution are in essence \( K \)-dimensional vectors of probabilities, which parameterize a categorical distribution. When considered as a conjugate prior to categorical distribution, the easiest way to think about the concentration parameters \( \alpha \) is to see them as pseudo-counts, where \( \alpha_i \) is the number of times the outcome \( i \) has been observed as a sample from the categorical distribution. The counts are pseudo for two reasons: \( \alpha_i \) need not be an integer, and it does not represent data but prior knowledge.

Conjugacy implies that our posterior distribution over probabilities \( p(. \mid s,a) \) will also be a Dirichlet. Thus, we can represent our posterior over the transition dynamics \( p \) as \( |\mathcal{S}| \times |\mathcal{A}| \) \( \alpha \) vectors (also known as count vectors) of length \( |\mathcal{A}| \). Given the prior \( \alpha \) for \( p(. \mid s,a) \), the Bayesian update for the observation \((s' = s_t, a, s)\) is performed by simply incrementing \( \alpha_t \) by one. This gives us an efficient way to maintain a probability distribution over possible transition dynamics \( p \), and update this distribution with new data. However, inference of the model is only one aspect of model-based reinforcement learning. The agent must also choose a policy and make decisions according to it. Most importantly, the chosen policy impacts what data the agent gets to observe, which in turn impacts its posterior over the dynamics. Since the agent does not know \( p \), it cannot simply compute the optimal policy. Thus, it must behave in a way that gathers data to learn \( p \), while trying to maximize its optimality criterion. For instance, it might be the case that an action \( a \) in state \( s \) does not provide any reward, but reveals the transition dynamics \( p \) entirely, so that the agent can simply compute its optimal policy after playing \( a \) once. This tension between information gathering and optimization is often called the exploration-exploitation tradeoff.
tradeoff in reinforcement learning [19]. Fortunately, Bayesian model-based RL provides us a model, namely Bayes-adaptive MDP (BAMDP), which once solved gives us the optimal policy for balancing this tradeoff [20, 21].

Given an MDP \( M = (\mathcal{S}, \mathcal{A}, p, r, H) \) where \( p \) is unknown to the agent, a Bayes-adaptive MDP is defined as the tuple \( M^+ = (\mathcal{S}^+, \mathcal{A}, p^+, r^+, H) \) where \( \mathcal{S}^+ = \mathcal{S} \times \mathcal{H} \) is the augmented set of states with \( \mathcal{H} \) denoting the set of all possible histories. With the inclusion of \( \mathcal{H} \), an augmented state \( s^+ \in \mathcal{S}^+ \) can represent both the current environment state \( s \) and the agent’s epistemic uncertainty about the dynamics in the form of a probability distribution over \( p \). This is simply because given a prior, the agent’s posterior is fully determined by the history it has observed so far. Let \( P(p \mid h) \) denote the agent’s distribution over the transition dynamics \( p \), where \( h \) is a history. For the discrete cases, the agent’s posterior \( P(p \mid h) \) is a Dirichlet. Thus, in practice the history \( h \) is replaced by the Dirichlet parameters \( \alpha \), where augmented states are \( s^+ = (s, \alpha) \). However, for generality, we will maintain \( h \) in the formulation. The augmented transition dynamics of the Bayes-adaptive MDP is,

\[
p^+((s', h') \mid (s, h), a) = \mathbb{I}[h' = h \Rightarrow p(s'|s, a)]; \tag{2.5}
\]

where \( \mathbb{I} \) is the indicator function. Intuitively, this means if the agent’s current belief about the dynamics is represented by \( P(p \mid h) \), then it uses the expected transition dynamics with regard to its belief for modelling the one-step transition probabilities. The indicator function ensures that the transition from \( h \) to \( h' \) tracks the correct history. We have made an important conceptual step here, which deserves attention. A Markov decision problem represents a decision-making task, where once solved, gives us a policy that maximizes an optimality criterion based on our cumulative reward. On the other hand, the Bayes-adaptive MDP represents both a learning and a decision-making task: the problem it models requires both learning about the dynamics \( p \) and maximizing. The reward function of the BAMDP is the same as the original MDP, such that \( r^+((s, h), a) = r(s, a) \).

The BAMDP dynamics defined in equation 2.5 is fully known to the agent, even though the MDP dynamics \( p \) is not known. The BAMDP can be treated as any MDP, thus planning methods or dynamic programming can be used to solve it. For instance, we can compute the optimal Q function \( Q^+((s, h), a) \) for the BAMDP, and simply apply \( \arg \max \) to recover the optimal policy. Since our states include posteriors over dynamics, the \( Q^+((s, h), a) \) is computed by considering every possible future transitions. Thus, effectively, this computation integrates over every possible future trajectory and corresponding posterior updates. This results in a policy that chooses actions by balancing how much reward they provide with how much information they provide about the dynamics \( p \). Such policies are called Bayes-optimal, since they provide an optimal balance of exploration and
exploitation with respect to the prior, in terms of Bayesian decision theory. Unfortunately, solving a BAMDP exactly is computationally prohibitive. This is due to the fact that if $H = \infty$, the set of states is countably infinite since it contains every possible history. For finite horizon problems, we have finitely many histories, however the set of possible histories is prohibitively large. This is because, since we do not know $p$, we cannot exclude any transition. Thus, if there are $|\mathcal{S}|$ states and $|\mathcal{A}|$ actions, for a horizon of $H$, there are $|\mathcal{S}|^H \times |\mathcal{A}|^H$ possible histories.

**Bayes-adaptive Monte Carlo Planning** [22]. Given that solving BAMDPs exactly is computationally infeasible, what should we require from an approximate algorithm? We would want an algorithm that can run for a specified amount of time, and provide a good enough solution. Additionally, when run for long enough, it should reach the optimal solution. Such planning algorithms are often called *anytime*, since the planning can be stopped at any time. Bayes-adaptive Monte Carlo Planning (BAMCP) is a sample-based anytime planning algorithm for solving BAMDPs, based on the Monte Carlo Tree Search (MCTS) [22].

The BAMCP is, by our definition, a model-based algorithm which uses a sampling model, i.e. a simulator. In essence, the algorithm consists of two procedures: *search* and *simulate*. Let us assume we start our planning at $s^+ = (s, h)$. The algorithm starts by sampling a transition dynamics $\tilde{p} \sim P(p|h)$ from the posterior distribution induced by the history $h$. Then the simulation starts from $s^+$, using the $\tilde{p}$ as a sampling model to simulate transitions. The simulate procedure is in essence identical to applying Monte Carlo Tree Search to the BAMDP, except the tree nodes are maintained for augmented states. This means paths along the tree correspond to trajectories (i.e. possible future “histories”), and each node can represent the posterior distribution induced by its trajectory. In essence, the only place BAMCP differs from directly applying MCTS to BAMDPs is the fact that a transition dynamics model $\tilde{p}$ is sampled before an iteration of simulation, and assumed fixed until the simulation round is over. This sampling approach is called *root sampling*, since a dynamics model is sampled and then fixed at the root of the search tree. The search procedure continues until timeout, which can also be defined as until a certain number of samples $\tilde{p} \sim P(p|h)$ are used. It is easy to see that the sampling approach in the search procedure is a Monte Carlo estimation for taking the expectation with regard to the current posterior. Additionally, if the simulate procedure is run for a long enough horizon and enough times, every part of the tree is visited infinitely often. This together means given enough compute, BAMCP converges to the exact solution of a BAMDP. For the proof of this statement and the details of the algorithm, we refer to [22].

When the loop inside the search procedure terminates, it leaves us with a populated search tree, where the root node maintains the estimated
state-action values $Q(s^+, a)$ for $s^+$. Then, the agent simply performs an argmax, and performs this action in the real environment. Once the new real transition is observed, $s^+ = (s, h)$ is updated accordingly, and the search procedure is repeated from the new state. Therefore, BAMCP is an online planning algorithm which computes $Q$-values for the current state, instead of solving the BAMDP for every state beforehand, which is infeasible.

### 2.4 Partially-observable Markov Decision Process

Up to this point, we have assumed the agent can observe the true state of the world $s$. However, in many real-world applications, the $s$ may be hidden from the agent, and they may only have access to noisy observations of it. This is called partial observability, and modelled with partially-observable MDPs (POMDP) [23, 24]. Mathematically, a POMDP is defined as $(\mathcal{S}, \mathcal{A}, \Omega, p, O, r)$ where the $\mathcal{S}$, $\mathcal{A}$, $p$, and $r$ are defined similarly to MDPs. Unique in POMDPs, $\Omega$ denotes the set of observations, the signals the agent receives instead of the states, and $O(o | s', a)$ denotes the probability of observing $o \in \Omega$ when the agent arrives at $s'$ by taking the action $a$. An important point to notice is that the POMDPs are a strict generalization of MDPs. If $|\Omega| = |\mathcal{S}|$, we can always recover the definition of an MDP by having a unique observation for each state with an observation probability of 1 at the state.

In POMDPs, the agent's history consists of action-observation pairs, such as $h_t = (a_0, o_0, ..., a_{t-1}, o_{t-1})$, since it does not have access to the states. For MDPs, there always exists a Markovian optimal policy, which means current state $s$ is a sufficient statistic for optimal decisions. Unfortunately, there is no such guarantee for POMDPs, since the observations do not maintain the Markov property. The existence of a deterministic optimal policy still holds though, so we must consider the class of history-dependent deterministic policies $\Pi_{HD}$. Fortunately, the agent does not need to remember its entire history $h_t$, but can compute a sufficient statistic of it. This statistic is called the state belief $b(. | h_t) \in \Delta(\mathcal{S})$, which represents the agent's state uncertainty. The $b(. | h_t)$ can be computed recursively as:

$$b(s' | h_{t+1}) = \frac{O(o_t | s', a_t) \sum_{s'' \in \mathcal{S}} p(s' | s, a_t) b(s)}{\sum_{s'' \in \mathcal{S}} O(o_t | s'', a_t) \sum_{s'' \in \mathcal{S}} p(s'' | s, a_t) b(s)}.$$  \hspace{1cm} (2.6)

The equation 2.6 is essentially a Bayes filter. When the agent takes action $a_t$ and observes $o_t$, it updates the agent's belief about the environment's state using $O$ and $p$, where the agent's current state belief is used as prior. Most importantly, $b(s' | h_{t+1})$ is a sufficient statistic for optimal policies in a POMDP [25]. Therefore, history-dependent policies can be express as $\pi_t(. | b)$ for POMDPs. Since the belief can be maintained recursively, we will drop the explicit dependency on $h_t$, and simply write $b_t(s)$ to denote the agent's state belief at time-step $t$. 

31
We can replace the state $s$ with belief $b$, and re-write the MDP optimality criteria for POMDPs as follows:

$$V^*_t(b) = \max_{a_t \in A_t} \mathbb{E}_{s \sim b}[r(s, a_t) + \lambda \sum_{s' \in \mathcal{S}} p(s' | s, a_t) \sum_{o \in \Omega} O(o | s', a_t) V^*_{t+1}(b')]$$

(2.7)

where $b'$ denotes the updated belief, when $b$ is updated with a new action-observation pair $(a_t, o)$.

Various model-based and model-free reinforcement learning methods can be applied to POMDPs. For instance, it is clear from equation 2.7 that value iteration can be applied in the space of beliefs. Since the space of beliefs is infinite-dimensional, approximate value iteration approaches specific for POMDPs have been developed (e.g. point-based value iteration [26]), which take advantage of certain structural properties of POMDP value functions. Other approaches include policy search with a restricted class of policies and heuristic search [25]. Recently, deep learning methods also have shown some empirical success in performing reinforcement learning in POMDPs [27, 28, 29, 30, 31].

**Bayesian Model-based RL for POMDPs and Bayes-adaptive POMDPs.** Just like for MDPs, Bayesian model-based approaches to POMDPs start with placing priors. In the case of discrete $\mathcal{S}$, $\mathcal{A}$, and $\Omega$, the observation probabilities $O(\cdot | s', a)$ induce a categorical distribution just like the $p(\cdot | s, a)$. Therefore, if both $O$ and $p$ are unknown, Dirichlet priors can be placed on them just like in BAMDPs. However, Bayes-adaptive POMDPs (BAPOMDP) have an important difference from BAMDPs. Since the agent cannot observe the states, it cannot know which transitions took place. In other words, the agent never gets to see samples from $p$. Additionally, since the agent is uncertain about the transition, it cannot be certain about which next state $s'$ to use for conditioning $O(\cdot | s', a)$. This means that there are multiple posterior distributions consistent with an action-observation history $\langle a_0, o_0, .., a_t, o_t \rangle$, depending on which latent transitions have actually occurred. A natural Bayesian way to deal with this uncertainty is to use hierarchical modelling, where we maintain a posterior distribution over posteriors. As we define in the next paragraph, Bayes-adaptive POMDPs achieve this elegantly, by extending the belief over environment states of an MDP, $b \in \Delta(\mathcal{S})$, to belief over augmented states $b \in \Delta(\mathcal{S}^+)$ of a BAMDP.

Mathematically, a Bayes-adaptive POMDP (BAPOMDP) is defined by the tuple $(\mathcal{S}^+, \mathcal{A}, \Omega, p^+, O^+, r^+)$ where the $\mathcal{A}$ and $r^+$ are defined similarly to BAMDPs. However, this time $\mathcal{S}^+$ includes the count vectors for two Dirichlet distributions: one over transition dynamics $p$ denoted by $\alpha$, and one over observation probabilities $O$ denoted by $\beta$. The augmented transition dynamics of the BAPOMDP is

$$p^+((s', a', \beta') | (s, \alpha, \beta), \alpha) = \mathbb{I}[\alpha' = \alpha + \delta_{ss'}] \mathbb{I}[\beta' = \beta + \delta_{\beta o}]$$

$$\mathbb{E}_{P(p|\alpha)}[p(s' | s, \alpha)] \mathbb{E}_{P(O|\beta)}[O(o | s', a)]$$

(2.8)
where \( \mathbb{E}_{P(p|a)}[\rho(s'|s,a)] \) is the expected transition dynamics under the Dirichlet distribution parameterized by \( \alpha \) and the \( \mathbb{E}_{P(O|\beta)}[O(o|s',a)] \) is the expected observation probabilities under the Dirichlet distribution parameterized by \( \beta \). The vectors \( \delta_{ss'}^{a} \) and \( \delta_{s_o}^{a} \) are count vectors that are zero everywhere, except for the transition \((s',a,s)\) and observation \((o,s',a)\), in which case they are 1. In essence, the terms \( \|a' = \alpha + \delta_{ss'}^{a}\| \) and \( \|\beta' = \beta + \delta_{s_o}^{a}\| \) make sure the count vectors are incremented correctly within the BAPOMDP transitions. Since both the observation and transitions are taken into account in the BAPOMDP’s dynamics model \( p^+ \), the observation model is deterministic and given as follows

\[
O^+ (o,|(s',a',\beta'),(s,\alpha,\beta),a) = \|[\alpha' = \alpha + \delta_{ss'}^{a}][\beta' = \beta + \delta_{s_o}^{a}].
\] (2.9)

In order to explain how the belief update in a BAPOMDP works, let us start from a prior belief \( b_0(s^+ = (s,\alpha,\beta)) = b_{0,s}(s)\|[\alpha = \alpha_0]\|[\beta = \beta_0] \), where \( b_{0,s} \) is a prior belief over environment states, and the \( \alpha_0 \) and \( \beta_0 \) are the count vectors for our Dirichlet priors on \( p \) and \( O \). Thus, our prior belief \( b_0 \) only has uncertainty over the environment states, and not the count vectors. Once our agent takes an action \( a \) and receives observation \( o \), it will update its belief according to the following equation:

\[
b((s',a',\beta') | a, o) = \frac{O^+ (o,|(s',a',\beta'),(s,\alpha,\beta),a) \sum_{s'\in S} p^+ ((s',a',\beta') | (s,\alpha,\beta),a) b_0(s^+)}{Z},
\] (2.10)

where we omit writing out the normalizing constant \( Z \) for clarity. What is important to notice here is, since \( b_0(s^+) = b_{0,s}(s)\|[\alpha = \alpha_0]\|[\beta = \beta_0] \), we practically condition on \((s,\alpha_0,\beta_0)\) as the current BAPOMDP state, where only the environment state \( s \) is a random variable. However, due to the structure of \( p^+ \) and \( O^+ \), even though we start with no uncertainty over count vectors, once the belief is updated, we may have more than one \((a',\beta')\) pairs consistent with the history \((a, o)\). Thus, the belief \( b((s',a',\beta') | a, o) \) will place non-zero probability on multiple Dirichlet posteriors for \( p \) and \( O \).

Since we do not know at which \( s \) we truly are, and also do not know which \( s' \) we have transitioned to, we cannot be certain of how to increment \( \alpha \) or \( \beta \). This uncertainty due to partial observability also takes away the general learnability result of BAMDPs, where we are guaranteed to converge to the true transition probabilities in the limit of infinite data.

Just like in the BAMDPs, solving a BAPOMDP gives us the Bayes-optimal policy in terms of balancing the exploration–exploitation tradeoff. However, this exact solution is even harder to achieve compared to BAMDPs. Infinite-horizon problems can be reduced to finite-horizon ones with a guarantee of \( \epsilon \)-optimal policies [32]. Fortunately, the BAMCP algorithm was extended to BAPOMDPs, in which case it is called the Bayes-adaptive Partially-observable Monte Carlo Planning (BAPOMCP)
Unlike the root sampling BAMCP, this algorithm’s root sampling variant samples a state $\tilde{s}^+ = (s, \alpha, \beta)$ from the current belief $b$. Then, it samples a transition dynamics $\tilde{p}$ and an observation function $\tilde{O}$ from the Dirichlet posteriors parameterized with $\alpha$ and $\beta$. The dynamics and observation function are fixed within a simulation round, as in BAMCP. For the details of the algorithm, we refer to [33].

Above, we have provided a brief overview of single agent reinforcement learning and the Bayesian model-based approaches to it. The BAMDP and BAPOMDP models will be the backbone of the multi-agent models we use in our contributions to human-AI collaboration. Most importantly, we will show that in some cases, the multi-agent problems will reduce to special classes of these models, where the classes have specific structural properties.

### 2.5 Multi-agent Reinforcement Learning and Human-AI Collaboration

Multi-agent reinforcement learning generalizes aforementioned reinforcement learning models to settings where there are more than one agent acting in an environment, making decisions. This difference has major implications, and makes learning harder. When describing a multi-agent scenario, there are many dimensions we must take into account. First, we must consider whether agents are independent and self-interested or not. In the case of human-AI collaboration, this assumption will always hold, since we do not have any control over the human’s behaviour. Another dimension we must establish is whether the setting is general-sum, zero-sum, or fully-cooperative. Zero-sum settings are fully competitive, since the rewards of agents are diametrically opposed: increasing one decreases the other. This is clearly not suitable for human-AI collaboration. In general-sum settings, agents’ individual rewards are different but not diametrically opposed. Thus, their rewards may in some cases encourage cooperation, and in others not. In fully-cooperative settings, the agents will have precisely the same reward or objective, thus should always cooperate. In this thesis, we will consider either fully-cooperative or general-sum cases. More specifically, in publication I and II, the human and the AI will have the same reward. On the other hand, in publication III, even though they have the same base reward, they will have different cost terms added.

In the following, we will start by defining multi-agent generalizations of the single agent reinforcement learning models we have described before. While doing so, we will touch upon the added difficulties of learning to collaborate in multi-agent settings, such as non-stationarity. The models we describe are not specific to human-AI collaboration, but the fact that the other agent is a human determines which modelling assumptions are
sensible and which are not.

2.5.1 Partially-observable Stochastic Games

A partially-observable stochastic game (POSG) generalizes POMDPs to the multi-agent setting [34]. It is defined by the tuple $\langle N, S, \{A_i\}, \{\Omega_i\}, p, (R_i), \mu \rangle$ where $N$ is a set of agents indexed by $i = 1, \ldots, |N|$, $S$ is a set of states, $\{A_i\}$ is a collection consisting of each agent’s set of actions, and $\{\Omega_i\}$ is a collection of the observation sets of the agents. We define the joint set of actions as $A = \times_{i=1}^{|N|} A_i$ and the joint set of observations as $\Omega = \times_{i=1}^{|N|} \Omega_i$. The joint transition-observation probabilities are given by $p(s', o | s, a)$ such that $o \in \Omega$ and $a \in A$. Each agent has their own reward function $R_i : S \times A \rightarrow \mathbb{R}$, and finally the $\mu \in \Delta(S)$ is the initial state distribution.

There are important differences between a POSG and a POMDP, which must be emphasized. Most importantly, the transition and observation probabilities $p(s', o | s, a)$ depend on the joint action $a \in A$. To illustrate the implications of this, imagine we have two agents indexed by $i$ and $-i$. The notation $-i$ is common in multi-agent learning, which means all agents but $i$. Let us take the subjective perspective of the agent $i$, who is trying to decide which action to take at time-step $t$. Unfortunately, unless it knows what the agent $-i$ is going to do from $t$ onwards, it cannot compute an optimality criterion similar to expected sum of rewards. This is because to use the transition and observation probabilities $p$, it must condition on both agents’ actions. However, if the agent $i$ had a way to know exactly what $-i$ is going to do in every possible history, and also had access to the observations of $-i$, the POSG would reduce to a POMDP in its perspective. Therefore, we can argue from the perspective of the agent $i$, the unique difficulty of the multi-agent setting is that it needs to be able to predict how others will behave.

Unlike in single agent models, we have not yet defined an optimality criterion for POSGs. This is because unlike in single agent cases, there is no consensus on what is a good optimality criterion in multi-agent learning. Since each agent have their own rewards $R_i$ and they are self-interested, we might think that each agent maximizing their own reward independently of each other is a good criterion. Learning algorithms that are based on this independence assumption such as the independent Q-learners do not have convergence guarantees, and can perform poorly due to non-stationarity [35]. Game theory offers many optimality criteria such as Nash equilibrium [36], thus we may choose that the agents must strive for reaching a joint policy that maintains a Nash equilibrium. However, there exists many Nash equilibria that are significantly worse than alternatives. For instance, in a game as simple as Prisoner’s Dilemma, the only Nash equilibrium is also the worst outcome possible for everyone.

Game-theoretic analysis can be seen as an objective perspective to multi-
agent systems. The optimality criteria it provides often look at the system as a whole, instead of from the perspective of a specific agent. Therefore, in many cases, game theory studies what are the interesting joint policies agents may find themselves in, but it rarely studies how the agents may reach this joint policy. The complementary approach to the objective perspective is the subjective perspective approach to multi-agent systems [37]. This approach looks at the multi-agent system from the perspective of a protagonist agent. In this thesis, we follow this approach and since the AI is the only agent we have control over, we take the AI’s perspective in human-AI collaboration. From now on, we will use i to denote the protagonist (AI), and −i to denote the other agent (human).

2.5.2 Bayesian Best-response Models

As described above, if i had a way to predict the other agent −i’s behaviour accurately, we could reduce the POSG to a POMDP, in which case the solution to this POMDP would be i’s optimal policy. However, in human-AI collaboration, it is highly unrealistic to assume that an AI can predict the human’s behaviour accurately beforehand. Therefore, we need a model where i learns about −i’s behaviour in addition to the transition and observation probabilities.

In Bayesian model-based reinforcement learning, we learn the transition and observation probabilities p by essentially putting a prior over a space of models for p. Since we know that the conditional probabilities p(s’,o | s,a) make up a categorical distribution, the model space is indeed the space of categorical distributions. Similarly, in order to learn about −i’s behaviour, we need to have a model space for it. The BA-BRMs model the agent −i’s behaviour as m−i = (A−i,Ω−i,I−i,β−i,π−i) where the A−i and Ω−i are the action and observation sets of −i, I−i is a set of internal states, β−i : I−i × A−i × Ω−i → Δ(I−i) is a belief update function, I−i is the current internal state, and π−i : I−i → Δ(A−i) is the −i’s policy [38]. Intuitively, the internal states I−i are how the agent internally represents its current state, and its policy depends only on the current internal state I−i. The function β−i represents how the agent updates its belief over the internal states, with new observations. Except for the I−i, all parts of the model are static, which are often called the frame of the agent, f−i, where m−i = (f−i,I−i).

The m−i is quite expressive, in the sense that it can capture many types of agents by specifying its frame. For instance, we can model a POMDP agent by choosing I−i as the space of possible state beliefs, where each I−i is a distribution over the states of the POMDP. In this case, β−i would map the previous belief to the next belief according to the update rule in equation 2.6. On the other hand, the agent we want to model may in fact use a weighted particle filter, where instead of the full state belief, it maintains a set of particles. Here, the normalized weights of each particle
form a discrete probability distribution. This case can easily be modelled by choosing $\mathcal{I}_i$ as the set of possible particle sets, and $\beta_{-i}$ as the filtering procedure that updates particle weights. In fact, as discussed in [38], the model $m_{-i}$ can express many styles of policies, such as finite-state controllers or policies of an agent who is learning and adapting to $i$’s behaviour. Because in the end, we can define $\mathcal{I}_i$ as the set of all possible histories, which include what $-i$ has observed about $i$’s behaviour. Then an internal state $I_{-i}$ would include both the environment and $i$’s past behaviour. If a policy is allowed to condition on $I_{-i}$, it can adapt to $i$’s behaviour.

Let us imagine we are in a POSG, where at time $t$, our agent $i$ is given the true frame of the agent $-i$, denoted by $f^*_i$, yet we are uncertain about the current internal state $I_{-i}$. Then, we can construct our model space for $-i$ as $\mathcal{M}_{-i} = (f^*_i) \times \mathcal{I}_i$. In fact, if there are other unknown elements about the $-i$ from its frame, we can include those into the internal state space when constructing $\mathcal{M}_{-i}$. This is indeed our approach in publication III. Once we have this model space, we can construct an augmented POMDP that provides the best policy for $i$ when paired with $-i$ (i.e. the best-response of $i$). Let us define the multi-agent environment of $i$ as $\text{MAE}_i \triangleq (\mathcal{N}, \mathcal{I}, [\mathcal{A}_k \forall k \in \mathcal{N}], \{\Omega_k \forall k \in \mathcal{N}\}, p, R_i, \mu)$, which is essentially the POSG, except only the $i$’s reward is considered. Together with $\mathcal{M}_{-i}$, the $\text{MAE}_i$ induces a best-response model defined as $\text{BRM}_i(\text{MAE}_i, \mathcal{M}_{-i}) \triangleq (\mathcal{F}, \mathcal{A}_i, \Omega_i, \bar{p}_i, \bar{R}_i)$ where $\mathcal{F} = \mathcal{I} \times \mathcal{I}_{-i}$, and the $\mathcal{A}_i$ and $\Omega_i$ are the sets of actions and observations for the agent $i$, the same as from the definition of a POSG. Once solved, this model provides the $i$’s best-response to agents that are from $\mathcal{M}_i$, and its dynamics are defined as

$$\bar{p}_i(\bar{s}', o_i | \bar{s}, a_i) = \sum_{a_{-i}} \sum_{o_{-i}} p(s', o | s, a) \prod_{-i} \beta_{-i}(I_{-i}^* | I_{-i}, a_{-i}, o_{-i}) \pi_{-i}(a_{-i} | I_{-i}) \tag{2.11}$$

and the reward function $\bar{R}_i$ is defined as

$$\bar{R}_i(\bar{s}, a_i) = \sum_{a_i} R_i(s, a) \prod_{-i} \pi_{-i}(a_{-i} | I_{-i}). \tag{2.12}$$

The $\text{BRM}_i$ is in fact a POMDP, therefore the best-response models of $i$ can be thought of as a special class of POMDPs. We can pair them with any optimality criterion suitable for POMDPs, and apply the same solution algorithms. But perhaps more interestingly, the fact that the $\text{BRM}_i$ is a POMDP opens up the possibility of learning $\bar{p}_i$ interactively with Bayesian model-based reinforcement learning. The implications of this are different from the single agent case. This is because the dynamics of the $\text{BRM}_i$ is in fact determined by two factors: the dynamics $p$ of the POSG, and our model $m_{-i}$ of the other agent. This implies that learning $\bar{p}_i$ includes also learning about the $-i$’s behaviour. In fact, as in publication III, for many
realistic cases we already know the dynamics of the task environment \( p \), but we do not know about the other agent’s behaviour. In this case, Bayesian model-based RL for \( \tilde{p}_i \) is the same as learning a model of \(-i\) interactively, while trying to best-respond to their behaviour.

Similar to BA-POMDPs, a Bayes-adaptive Best-response Model (BA-BRM) for the agent \( i \) is defined as a tuple \((\mathcal{S}^+, \mathcal{A}_i, \mathcal{O}_i, \tilde{p}^+_i, R^+_i)\), where the states now include the count vectors for transition and observations as \( \tilde{s}^+ = (s, I_{-i}, a) \) where \( a_{s,a}^{\bar{s}_i} \) is essentially the count for the event of \((s', o_i)\) happening after action \( a_i \) is taken at \( s \). The augmented dynamics \( \tilde{p}^+_i \) are defined as

\[
\tilde{p}^+_i ((s', I_{-i}, a'), o_i | (s, I_{-i}, a), a_i) = \mathbb{I}[a' = a + \delta_{a_i}^{\bar{s}_i}] \mathbb{E}_{\tilde{p}(\tilde{p})} [\tilde{p}(s', o_i | \tilde{s}, a_i)], \tag{2.13}
\]

which is a special class of the BA-POMDP dynamics defined in equation 2.8, and includes the observation probabilities as well as the transition dynamics. Since this is a BA-POMDP, the algorithms such as BA-POMCP can be directly applied to it.

In essence, BA-BRMs formalize a fundamental intuition: from the perspective of the agent \( i \), the other agent \(-i\)’s behaviour in a POSG matters because of its impact on the transition and observation probabilities. If we have an expressive enough model space for \(-i\)’s behaviour, we can infer it from its influence on the dynamics, which is essentially the same as learning the dynamics model. Even if the agent \(-i\) is a complex learning algorithm that adapts to our policy, there is no reason why its behaviour cannot be learned as long as our definition of its internal state space is rich enough.

### 2.6 Conclusion

In this chapter, we have provided an overview of single and multi-agent reinforcement learning, with an emphasis on models and model-based approaches. All the comprising articles make use of the models and algorithms described in this chapter. In the end, our methodology for using model-based approaches in human-AI collaboration is as follows: First, we identify the common nature of the tasks we are facing, and what are the reasonable modelling assumptions we can make about the human. For instance, if the AI’s task is to help the human build a statistical model, we may assume that the human user’s statistical knowledge will influence their behaviour. Then, we can choose to define the internal states \( \mathcal{S}_{-i} \) to represent possible statistical knowledge levels, as in publication II. Alternatively, in a long-term decision-making task, we may expect the human to make mistakes due to time-inconsistency bias (described in section 3.2). Then, we can choose to model time-inconsistent behaviour by using its mathematical parameterizations provided by cognitive science.
and behavioural economics, as in publication III.

Once we have built a parameterized model for realistic human behaviour in our task, the Bayesian models provided in sections 2.5 and 2.2 provides us the necessary decision-theoretic framework to learn what is a good policy, which allows both learning about the human’s behaviour and optimizing our optimality criteria. Since we consider human-AI collaboration, the AI’s optimality criteria are always to improve the human’s decision-making performance.
3. Contributions

In this chapter, we summarize the contributions of each comprising article.

3.1 A Machine Teaching Framework for Human-AI Collaboration (Publications I and II)

We will start providing an introductory background for machine teaching, and then present a summary of the contributions of publications I and II. A brief background on models and methods used here is presented in section 2.2. Further details can be found in the manuscripts of publication I and II.

3.1.1 Machine Teaching

In essence, machine teaching represents the inverse problem to machine learning. Given a dataset $\mathcal{D}$, the goal of a machine learning algorithm is to find the best model possible according to a specified criterion, such as minimizing a loss function. Thus, we can imagine the overall procedure of machine learning as one that receives a dataset as input and outputs a model. Then, the inverse of this procedure should receive a model and output a dataset. Indeed, the main question of machine teaching is: Given a target model $\theta^*$ and a machine learning algorithm $Alg$, what is the optimal dataset $\mathcal{D}^*$ that makes $Alg$ learn $\theta^*$? Two main criteria for optimality are: (I) Find the smallest dataset possible (i.e. minimum $|\mathcal{D}^*|$) that produces $\theta^*$, or (II) Have the $Alg$ output a model $\theta$ as close as possible to $\theta^*$ (i.e. minimize $d(\theta, \theta^*)$ with respect to a discrepancy measure $d$) with a fixed number of data points $|\mathcal{D}^*| = K$. Machine teaching provides us with interesting theoretical questions and tools. For instance, it can be used to measure the complexity of different machine learning problems or hypothesis classes. Indeed, teaching complexity proposed by [39] measures how difficult it is to teach specific learners different concepts, and uses this as a complexity measure for the learning setting itself.
If the machine learning algorithm $Alg$ can be trained sequentially, the machine teaching of $Alg$ can be formulated as an optimal control problem, where the system to control is the $Alg$ [40]. For example, if $Alg$ is a Bayesian inference algorithm, it will output a posterior $p(\theta | \mathcal{D}_t)$ where $\mathcal{D}_t$ denotes the data it had received so far. Given this output, we can feed new data points into it, and at the next time-step it would output $p(\theta | \mathcal{D}_{t+1})$. As a matter of fact, the next posterior depends only on the current posterior, the current data, and the new data points we input. Then we can construct an MDP where the actions are data points, states consist of posterior distributions, and the transition dynamics are governed by $Alg$. In practice, we are often given a dataset $\mathcal{D}$, and try to find the best subset of data from it, which constraints the action space to $\mathcal{D}$. The teaching problem becomes finding the optimal sequence of actions that reaches a goal state. Here, we can represent the second optimality criterion of machine teaching as a finite-horizon MDP by minimizing expected sum of costs (equivalent to maximizing expected sum of rewards): The teacher wants to get as close as possible to a goal state $p^*$ in a finite horizon $H$, where it receives a terminal cost of $d(p^*, p_H)$. Solving this MDP exactly is computationally challenging, because it involves simulating a learning algorithm $Alg$ at each transition. Instead, we can use approximate methods such as MCTS with a random rollout policy: plan n-steps ahead, and then pick the rest of the data points uniformly randomly to approximate the Q values.

The publication I and publication II model different multi-agent settings as a sequential machine teaching problem. Moreover, the publication II extends the machine teaching framework to multi-objective and partially-observable settings.

### 3.1.2 Learning from Strategically-Steering Humans (Publication I)

In publication I, we have investigated whether a teacher–learner protocol can enable better AI assistance in interactive intelligent systems such as recommender and information retrieval systems. Previous approaches often model the human users of these systems as passive data distributors, where the feedback they provide to the system are considered as samples from a probability distribution that is parameterized by their preferences. However, with the increasing capabilities of intelligent systems, human users started extending their theory of mind reasoning to intelligent systems they interact with. This is often called the theory of AI’s mind (ToAIM)[41]. In certain cases, humans can use ToAIM to steer AI systems towards their desired behaviour. For instance, after publication I, recent work in recommender systems has emerged to show that humans provide recommender systems with active signals to steer the system behaviour [42]. An example of this is when users choose an item
that is the opposite of what is recommended, to *steer the system* away from similar recommendations in the future. Such behaviours cannot be fully accounted for as samples from the preference distribution, since they stem from the user’s strategic reasoning. Therefore, a new approach is necessary. In publication I, we answer the research questions RQ1.1 and RQ2.1 (see section 1.2) by modelling the user’s steering behaviour as an active attempt to *teach* the system their preferences. This means the interaction protocol between the intelligent system and the human can be modelled as machine teaching (as defined in 3.1.1), where the user is the teacher and the system is the learner. We empirically demonstrate that our approach enables better collaboration between the intelligent system and its user. Our results indicate the importance of interpreting active signals of users as such, instead of presuming all user behaviour is passive.

More specifically, the publication I models the human’s teaching behaviour as a reinforcement learning policy. When the human user gets item \( x \) recommended to them, they observe a reward, which essentially is determined by their preferences over items. As feedback, they may choose to pass on this reward directly to the AI, or change it. Thus, their feedback becomes the actions of their policy. Then here, the human user can compute Q-values (see equation 2.3) by simulating an interaction with the AI using their model of the AI. For instance, after getting recommended \( x \), the user can simulate what will they get recommended next if they give positive feedback for \( x \) many steps into the future, to compute approximate Q-values with methods similar to MCTS or other Monte Carlo approaches to reinforcement learning. In publication I, the strategic user chooses their feedback action with probability proportional to the approximate Q-value they computed, which is also known as the Boltzmann policy in control theory and as quantal response in behavioural economics [43, 44]. Considering not every user will behave like a strategic teacher, we use a linear mixture that combines this user type with the more traditional linear bandit model where the feedback to a recommendation is just a sample from a distribution with probability proportional to the reward of \( x \).

We evaluated the system’s performance with the mean cumulative reward of the AI, which measures the positive feedback it received from the user. This means to maximize this objective, the AI must have recommended the user what they prefer the most. Since the user’s reward function (i.e. preferences) are unknown to the system, this involves exploration. Our simulated results show that the ideal case is when the human user strategically teaches the AI by steering, and the AI models this correctly. The performance of our mixture model matched this case for strategic teaching, and also showed that the performance does not deteriorate if the user does not perform strategic teaching. In order to test our model with humans, we have designed and conducted a small-scale user
Contributions

study with 10 participants. The study represents an information retrieval task, where a user is looking for a particular document that they cannot recall but can recognize, and the AI’s job is to find the document. The average cumulative reward was higher with our mixture model compared to modelling the user as a passive distribution (i.e. linear bandit).

Related works in multi-agent learning. From the perspective of multi-agent learning for human–AI collaboration, the model we propose in publication I can be treated as an interactive POMDP (I-POMDP) in its most general case [45], and it is closely related to a variety of multi-agent models [46, 47, 48, 49]. These related models provide a general decision-theoretic framework to implement many other interaction protocols similar to our teacher–learner protocol. However, the I-POMDP model is almost all-encompassing, capable of capturing any multi-agent interaction. Thus, our model is a significant contribution in the sense that it identifies the specific structural properties induced by our interaction protocol, and take advantage of it. Additionally, I-POMDPs can represent recursive reasoning (I think that the other agent thinks that I think that...) up to arbitrary depth, which is akin to theory of mind in humans, which enables better social collaboration [50, 51]. In the case of publication I, the AI’s model of a teaching human has in turn a model of the AI (i.e. ToAIM), which is recursive reasoning. Previous work in I-POMDPs has proposed the idea of learning from humans with recursive reasoning [52], yet our work is the first to propose a multi-agent recursive reasoning model in the practically important case of multi-armed bandits, allowing us to learn online from the scarce data emerging from human–AI interaction.

Related works in machine teaching. Previous work in machine teaching have theoretically studied learners who are aware of the teacher, similar to our model of the AI in publication I [53, 54]. However, in these settings, the teacher would design the entire training data for a learning algorithm, and was not allowed to alter a pre-existing data generating distribution. Since these assumptions were too strong for human–AI collaboration with human teachers, publication I relaxed both, by having the human provide sequential data interactively, and by removing the constraint that the data provided by the human teacher must be consistent with a pre-defined data distribution. Previous work in machine teaching mostly considers cases where the teacher computes the minimal subset of data needed from a given dataset, to teach the learner a target model [39, 55, 56]. This is different from our setting, where the teacher must instead compute a policy that provides the best sequence of data, rather than a batch. Therefore, our setting is most closely related to constructing batches of state-action trajectories for inverse reinforcement learners [57, 58]. Different cases of teaching gradient descent algorithms by providing them with a sequence of data points have also been considered [40, 59, 60]. For intelligent tutoring
of humans, teaching with uncertainty about the learner’s state has been formulated as planning in partially-observable Markov decision processes [61, 62]. Machine teaching can also be used towards attacking learning systems [63, 64] or modifying rewards online [65]. The goal, settings, and proposed methods differ from ours. The model of the teacher in publication I can be seen as an agent performing reward shaping, which aims to make the environment more supportive of the learner by alleviating the temporal credit assignment problem [66].

**Related works in user modelling.** User modelling aims at improving the usability and usefulness of collaborative human–computer systems by providing personalized user experiences [67]. For example, in information exploration and discovery, the system needs to iteratively recommend items to the user and update the recommendations based on the user feedback [68, 69]. The current underlying statistical models use the user’s response to the system’s queries, such as *did you like this book?*, as data for building a preference profile of the user. Recent works have investigated more advanced user models [70, 71]; however, as far as we know, no previous work has proposed statistical user models that incorporate a model of the user’s mental model of the system, who may be engaging in recursive reasoning.

### 3.1.3 Teaching Humans for Effective AI Assistance (Publication II)

When AI agents assist humans, the human users will always want to and should have the last say. However, this may become a hindrance for effective assistance. Most importantly, the human user may not have the necessary knowledge to interpret the assistance correctly, and decide to ignore or overrule the AI assistance even though it is helpful. In that case, an advanced AI system should try to teach the human user the knowledge needed for enabling effective assistance.

Publication II builds on the publication I by modelling the human user as a learning algorithm and the AI as a machine teacher. However, extending the machine teaching to apply to both human and machine learners require non-trivial advances. As explained in the following paragraphs, publication II presents a unifying model that treats both humans and machine learning algorithms as learners with dynamic internal states. This opens up a new direction of research by bringing together the intelligent tutoring community which focuses on designing algorithms to exclusively teach humans, and the machine teaching community that focuses mainly on teaching machine learning algorithms. Cross pollination between these two disciplines will allow for better computational models of teaching that take advantage of pedagogical models of humans.

In order to demonstrate how the user’s lack of knowledge may hinder
assistance, let us start with a running example. Imagine an AI assistant helping its user build a linear regression model to explore a dataset. An important decision in such statistical analyses is the choice of covariates to include in the regression. This is particularly true when there are too many covariates, and the inclusion of all leads to models that are difficult to interpret. It is safe to assume that a user with basic statistics knowledge would understand that if a covariate $x_i$ is strongly correlated to the output $y$, then it should be included. However, the user may fail to recognize collinearity, where one or more covariates are correlated among themselves. For instance, if $x_i$ and $x_j$ are perfectly collinear, then this means $x_i = wx_j$. Including such collinear covariates together into regression leads to identifiability issues in the weights. If we perform linear regression with both $x_i$ and $x_j$ included, since $x_i = wx_j$, any combination of regression weights $w_i, w_j$ that sums to the same constant will be the same. This makes interpreting the regression weights difficult. Under these circumstances, an AI assistant should recognize this and advise the user against including covariates that are collinear together. Unfortunately, if the user does not know about collinearity, they may choose to ignore this advice. If the AI can infer the user’s lack of knowledge from interaction, it can take actions to teach the user about collinearity. These actions would be costly, since they require more effort on user’s end. However, if it means the user will have better performance in their future regression tasks, this cost may be worth it.

Publication II mathematically formulates the intuitions described by the example above as machine teaching, similar to publication I. However, in this case, the teacher is the AI and the learner is the human user. Specifically, we model the human as a learning algorithm with an internal state $z$ that represents their inductive biases and prior knowledge. Therefore, we will use the terms user and the learner interchangeably. In current machine teaching methods, the internal state $z$ is assumed fixed, and in most of the cases, fully known by the teacher. In publication II, we generalize this to cases where the internal state is unknown, not fixed, and can be influenced by the teacher’s actions. Even though the internal state $z$ is latent to the teacher (e.g. the AI cannot directly observe whether the human knows about collinearity or not), the teacher can maintain a belief (i.e. probability distribution) over the set of internal states $Z$. This belief can be updated with observations that depend on $z$: for example, if the AI suggests the human a collinear variable and they accept it, this may suggest that the learner does not know about collinearity.

Mathematically, we describe the teacher’s task as a POMDP, where the set of states is a Cartesian product of the set of models the learner have, $\Theta$ (i.e. hypothesis space), and the set of its internal states $Z$. The set of actions consists of task-specific assistance actions (e.g. suggesting covariates) and tutoring actions (e.g. teaching collinearity). The assistance
actions can also be seen as actions that teach task-specific skills (e.g. teaching by doing). In contrast, tutoring actions teach high-level knowledge that makes it easier to teach task-specific skills, or provide task-specific assistance, to the user. Mathematically, the tutoring actions are defined as any actions that can trigger transitions in $z$, whereas task-specific actions change $\theta$.

Our theoretical results indicate that, if the learner is in an internal state $z$ that prevents them from receiving effective assistance (e.g. refusing to avoid collinearity), then the teacher must either hide data from the learner (e.g. not showing collinear variables to the user) or end up with a suboptimal result such as a linear model with collinear covariates. We have identified that the existing objectives of machine teaching reward the teacher only based on the learning result of the current task. In other words, machine teaching does not care about a learner’s independent performance or its performance on future learning tasks. This means existing machine teaching objectives have no incentive for tutoring, and would always prefer hiding data from the learner. This is especially disturbing when the learner is a human, due to obvious ethical reasons. In order to resolve this, we formulate a multi-objective approach, where the teacher's reward function for the POMDP is a linear combination of two terms: (O1) The traditional machine teaching objective depending on the goodness of the final model obtained for the current dataset and (O2) The learner’s independent performance in future datasets without a teacher, evaluated with held-out datasets. Since the teacher cannot assist the learner in the case of (O2), if the current internal state is unsuitable (e.g. learner does not consider collinearity), this performance will be low. Therefore, maximizing the second term incentivizes the teacher to induce transitions into better internal states by tutoring. If the weight for (O2) is set to zero, this recovers the standard machine teaching objective. On the other hand, setting (O1)’s weight to zero means teacher does not care about the current dataset at all, and chooses only to tutor. Thus, the multi-objective approach gives system designers considerable freedom.

The mathematical framework we have presented in publication II is in fact a novel method for applying machine teaching to meta-learning algorithms. Therefore, our framework bridges the gap between automated teaching of humans and of meta-learning algorithms. This presents opportunities for the machine teaching approaches to be applied to AI systems that teach and assist humans. Therefore, we evaluated our framework on both the AI assistance for covariate selection scenario and a setting where we teach an online meta-learning algorithm to perform few-shot regression for sine functions [72]. For the case of AI assistance, our empirical results demonstrate that if we ignore (O2) and only consider (O1), data manipulation is the optimal teaching solution and tutoring is never chosen. However, when both (O1) and (O2) are considered equally, tutoring
when needed is optimal since even though initially costly, it improves the learner’s independent performance in future tasks. For the case of teaching online meta-learners, our results demonstrate that our method can improve few-shot regression performance of learning algorithms by providing a better sequence of tasks to the meta-learner compared to a random sequence.

**Related works in machine teaching.** An iterative variant of machine teaching [59] assesses the iterative nature of some learning algorithms and shifts the problem from minimizing the size of a dataset to minimizing the number of steps. This method still assumes that the learner is fully-observed and the learning algorithm is known by the teacher, and that the teacher can only provide data points from the dataset. Closer to our framework, [73] alleviates these two problems, by considering that the learner and the teacher have different views of the same dataset and that the teacher does not know the algorithm of the learner, in the same way as [74]. Our framework differs from these methods, since they consider an unobserved but fixed algorithm for the learning, while our setting is built upon the possibility for the teacher to cause high-order changes (e.g. changes in inductive biases) in the algorithm of the learner. Also, we do not restrict the actions of the teacher to the choice of data points. This is especially useful for bridging the gap between intelligent tutoring of humans and machine teaching of learning algorithms, since a richer class of actions is available for teaching humans. In all the machine teaching works considered, the learner adapts to the teacher by updating only their estimated model and this line of work considers only the states of the world, whereas in our work we take one step further to considering the teacher’s influence on the internal states of the learner (e.g. its priors, learning rate, inductive biases...) which affects both the learned model and the learning algorithm.

**Related works in intelligent tutoring.** Intelligent tutoring of humans have been formulated in terms of MDPs and POMDPs before. For instance, [75] considers that the teacher uses an MDP to adapt its teaching policy to the learner during the teaching process. Multi-armed bandits has been suggested by [76] as a way to adapt to multiple types of learners. As an alternative, POMDPs have been proposed to alleviate the uncertainty over the learner’s cognitive state [61]. Unlike our method, these papers only consider adapting to various fixed types of learners, but do not consider the possibility of the teacher causing a change in the type of the learner.
3.2 A Bayesian Multi-agent Reinforcement Learning Framework for Human-AI Collaboration (Publications III)

Here, we present a summary of the contributions of the publication III. A brief background on models and methods used here is presented in section 2.2 and 2.5. Further details about the publication III can be found in the manuscript appended to the thesis.

3.2.1 Improving Human Intelligence with Centaurs

A centaur is an agent that is part-human, part-AI [77]. At the heart of the centaur concept is the following observation: computers and humans have complementary strengths and weaknesses, therefore they can cooperate to elevate each other. In other words, the AI’s goal in a centaur is to augment human intelligence by taking advantage of their complementary strengths. Publication III answers RQ1.2 and RQ2.2 (see section 1.2) by presenting the first mathematical model of a centaur, where the interaction between the human and the AI parts is modelled as a sequential game, and the AI must learn about its human partner through interaction. Our method presents a formal framework for bringing in prior knowledge from cognitive science and behavioural economics in the form of models of human behaviour, and takes advantage of these models in online decision-making. This opens up a new direction of research where different models from behavioural sciences can be used as inverse models within multi-agent reinforcement learning to develop AI agents that can augment human intelligence. The rest of this section goes into further details of our technical contributions.

As the designers of the AI part of a centaur, we do not have control over the human, neither should we. In this case, the AI we build must be able to infer certain properties of its human partner, and must help them make better decisions interactively. While doing so, it must never prohibit the human from having full authority. In publication III, we propose an interaction protocol where at every time-step $t$, first the AI proposes to take an action $a_m$, the human gets to observe this proposal and then must choose whether to accept it or override it. Accepting the action means the AI will be allowed to execute $a_m$ in the environment, but overriding it means the human will execute an action of their own instead. Conceptually, we can think of this as a human and an AI sharing a single controller that interfaces with the environment. In this interaction protocol, the human always have the last say.

As an illustrative example, consider a self-driving car with the human sitting in the driver’s seat. An AI agent is driving the car on a motorway, while the human is enjoying the view. The AI decides to change lanes, and indicates this using turn signals. The human disagrees with this
decision, and to override, takes the wheel to drive. In this example, the AI should interpret the human's decision to override correctly and learn from it. Importantly, the human preferred not to drive themselves until the AI made this particular mistake. This implies that, the perceived cost of letting AI change lanes was higher than the cost of driving for the human. Note that the running example used to explain publication II in section 3.1.3 closely resembles this setting. There, the AI proposes to add a covariate $x$ to the regression model, and the human can either accept this to allow the AI, or override this.

In the ideal case, the human and the AI would always agree on what to do, and thus the human never needs to override. For this to happen, the human and the AI must share similar models of the task environment, similar beliefs, computational bounds, and biases. However, humans can over or under-estimate risks, can have inaccurate models of the task environment, or may simply be constrained by their cognitive bounds. All of these can lead us to take objectively suboptimal actions, since they were optimal with regard to our models, bounds, and biases. Thus, differences between the human and the AI can lead to disagreements on what is the optimal action to take. Cognitive science and behavioural economics have been developing a rich variety of mathematical models in order to explain why humans make seemingly suboptimal decisions, under the field of computational rationality [78, 79, 80]. These models can be leveraged to infer sufficient statistics of human partners, and used to improve their decision-making performance [81].

In publication III, we propose a novel multi-agent model that mathematically formulates the intuitions above, using the subjective perspective approach to multi-agent systems [37]. We take the perspective of a specific agent, and model the AI and the human as having different subjective task models. Specifically, each agent's subjective task model consists of a POMDP $M_i$ and an optimality criterion $OPT_i$ as defined in section 2.2, where $i \in \{m, h\}$ with $m$ denoting the AI and $h$ the human. Therefore, human and the AI may differ in their estimation of transition and observation probabilities, prior state beliefs, and optimality criteria. These differences lead to disagreements on what is the optimal action to take, and can cause the human to override the AI. The human and the AI have the same reward function in their subjective models, therefore the task is fully cooperative. However, during interaction, if the human decides to override the AI, they get a cost term added to their reward. This cost is in no way enforced by the AI, and it is internal to the human. For instance, in the self-driving example, overriding means the human picking up the wheel and driving themselves. Compared to automating the driving, this will have additional mental and physical costs for the human.

Based on the subjective task models and the interaction protocol, we develop a model space for the human part of the centaur. We can start with
the observation that an optimal decision should weigh the additional cost of overriding the AI against the increase in reward, and override only if the latter is higher. This means the human’s policy is fully determined by their subjective task model, and the cost of overriding. The *machine-optimistic human model* presented in the publication III captures this decision rule for the case of a one-step lookahead approximation. In essence, this model is derived mathematically from assuming that human approximates the future optimistically, assuming they will never need to override the AI from $t+1$ onwards.

We can parameterize the space of possible human models we consider, using the mathematical models from cognitive science and behavioural economics. In publication III, we use two use cases to demonstrate the capabilities of our approach. For the first case, imagine that the human over-estimates the probability of failure for specific actions in transition dynamics. Such subjective models may lead to optimal policies that avoid these actions, even though they should not. This behaviour pattern is called *maladaptive avoidance* in behavioural sciences [82]. We can parameterize the space of the human’s subjective POMDPs with $\epsilon$ denoting how much the human over-estimates the true probabilities. If $\epsilon = 0$, this would mean no maladaptive avoidance.

The second case we use is called *time-inconsistency bias* [83, 84]. This bias is at the heart of many well-established deviations from perfect rationality in humans, such as procrastination, early stopping, and preference reversals. The simplest example of time-inconsistency is the dollar experiment. Here, people are asked if they would prefer receiving 100 dollars today, or 110 dollars tomorrow. Then, they are also asked if they prefer 100 dollars in 10 days, or 110 in 11 days. Interestingly, there is a considerable number of people who choose 100 dollars today, and 110 dollars in 11 days. Here, time-inconsistency means the preferences between two outcomes is not consistent across time, and depends not only on the delay (i.e. 1-day) but also on the actual time. Previous work have identified a mathematical model that is consistent with empirical evidence for time-inconsistency: hyperbolic discounting [85, 86, 87]. Here, we model the human as using the optimality criterion of expected sum of discounted rewards (see 2.2), where the discount function has the form $\lambda(t; \kappa) = \frac{1}{1 + \kappa t}$ and $\kappa$ is a parameter. If $\kappa$ is small enough, this function behaves very close to the exponential discounting. However, for large enough $\kappa$, it leads to time-inconsistent decisions and preference reversals. Therefore, in this case, $\kappa$ parameterizes the space of possible human models, where the subjective task model uses a hyperbolically-discounted optimality criterion for given $\kappa$.

We use Bayes-adaptive best response models (see 2.5) to model the AI’s problem of inferring the necessary parameters of the human’s subjective task model. In addition to the parameters of the human model space, the AI also does not know the cost of overriding for the human, and
must infer this. Our model space for the human induces a mathematical structure in the dynamics of the BA-BRM, and the AI takes advantage of the parameterization of the human model space to quickly infer the dynamics. Let us illustrate this with an example: consider a grid-world where whenever the AI wants to go up, the human overrides it with going right. This alters the subjective dynamics model of the AI, since for the AI, the action up transitions to right. Thus, for the AI, inferring the correct dynamics means inferring the human’s policy. This inference problem is made feasible by the parameterization of the model space of human policies. For instance, in this case, the AI may infer that the human over-estimates the probability of up actions failing by a large $\epsilon$. If the cost of overriding for the human is low, then AI may infer it will never be allowed to execute the action up. However, if the cost allows it, the AI may infer that up may be allowed.

In order to take advantage of the parameterization, we develop a novel adaptation of the root sampling BA-POMCP algorithm. In essence, we maintain a set of particles, each of which corresponds to a pair of a subjective task model and cost of overriding for the human. Since the human policy comes from solving their subjective task model, this is equivalent to maintaining a particle of human policies. For example, in the case of time-inconsistency, the particles consist of $\kappa$ and cost pairs.

We perform experiments for both time-inconsistency and maladaptive avoidance in simulated environments. For the former, we consider a case where the human’s preference reversal leads to decisions they regret afterwards. Our experiments demonstrate that the AI-part can quickly learn a policy that improves the human decision-making by avoiding states that lead to preference reversals if possible. If not possible, the AI learns to follow the exact policy desired by the human, albeit suboptimal. In case the human is not time-inconsistent, our method quickly learns the jointly optimal policy. In maladaptive avoidance, we consider both cases where the human or the AI over-estimates the failure probabilities of certain actions. Similarly to time-inconsistency, in case when the human is maladaptive, our method learns a policy that improves the human decision-making by taking the over-estimated actions when they would be accepted by the human. When the AI is over-estimating and the human’s model is correct, if the human is persistent with their overrides, our method learns that it must perform the policy the human wants, and overrides are not necessary afterwards.

In summary, our method and the Human-Machine Centaur (HuMaCe) framework presents a mathematically principled approach to using parameterized models of human decision-making in order to learn policies that will improve human intelligence. This can be seen as reinforcement learning how to nudge humans.
Related works. Previous works have shown that self-driving cars can leverage their influence on other human drivers to improve the traffic conditions for every one [88, 89, 90, 91, 92]. However, this line of work is complementary to ours, since it studies autonomous AI agents that try to influence the collective behaviour of humans, rather than semi-autonomous agents that try to influence the individual behaviour of their users. Therefore, our work aligns more with behaviour change support systems [93]. The multi-view MDPs and c-intervention games proposed by [94] are the closest models to ours. Here, the human and the AI’s subjective task models are MDPs. The subjective models disagree only by transition probabilities, and the human may override the AI’s action. Publication III generalizes this to partially observable environments and other types of disagreements between models (e.g., observation functions). Their interaction protocol is a Stackelberg competition, where the AI commits to a policy for the whole horizon, and the human observes the AI’s full policy before the beginning of interaction. Then, the AI is the leader, and the human is a follower who must best-respond to the AI’s policy. We relax these assumptions since in many cases, it is more realistic to assume the human will be able to observe the AI’s action, but not know its entire policy, and it may not be possible to make the human act as a follower. In addition, [94] assumes full knowledge of the human’s subjective task model, whereas our method infers this and the cost the human pays for overriding the AI. A similar yet less related model is the Helper Assistant MDP from [95], which only considers the case when the human’s goals are unknown and the AI can execute its actions without permission.

3.2.2 Theoretical Bounds of Improving Human Intelligence with Centaurs

In publication III, we identify a novel theoretical and practical problem for human-AI collaboration within a centaur, specific to partially-observable cases. To illustrate this, consider the Rock Sample task shown in figure
3.1. Here, the yellow robot must pick up good rocks (green) and enter the exit area. For every bad rock (red) it brings to the exit area, it is penalized, and for every good rock, rewarded. However, the quality of the rocks are only partially observable. The robot can measure any rock from any grid, by taking a measure-rock action. However, the further the rock $i$ is, the noisier the observations are, where the increase in noise is determined by a sensor sensitivity parameter $\epsilon$. Measurement actions also have a fixed cost associated to them. Now, let us consider the case where the centaur is controlling the robot. The human over-estimates $\epsilon$, and thus thinks the optimal policy is to go on top of each rock to get the best measurements possible. However, in reality, $\epsilon$ is low, and the AI estimates it correctly. Thus, the AI correctly computes the best policy to measure all rocks from the corner a couple of times, since this is much more time and cost-effective due to how the rocks are spread out. In this case, even if the cost of overriding was low enough to allow the AI to perform measurements from far away, convincing the human may be costlier than simply traversing all rocks. This is because the human is still updating their state belief using their subjective POMDP, according to the equation 2.6, which uses over-estimated $\epsilon$. Thus, given the exact same action-observation history, the AI and the human will have different state beliefs, which means they may continue disagreeing on the optimal action. We name this problem as the belief alignment problem. An important theoretical question to answer here is: what are the structural properties of the underlying observation and transition dynamics, which would allow the differing state beliefs of the human and the AI to contract? In publication III, we answer this question partially by deriving a contraction theorem for the beliefs, which depends on the mixing properties of the dynamics and the informativeness of observations.

The mathematical definition of the belief alignment problem and our early theoretical results indicate that further theoretical and structural research is needed to ensure human-AI collaboration in partially-observable tasks. An important future direction is to identify in what cases the belief alignment problem cannot be resolved by learning without explicit communication between the human and the AI. In such cases, we hypothesize that a prior belief elicitation phase, or explicit belief communication during the task will be necessary.

Related works in learning in games. This result can be seen as related to the problem of learning with misspecified models [96, 97], which focuses on characterizing how long-term beliefs of agents would evolve, when they misinterpret information due to misspecified models. However, our result is the first to investigate how the structural properties of the underlying transition and observation dynamics relate to disagreements in beliefs of different agents over time, even if they observe the same history.
3.3 A Call-to-arms for Advancing User Modelling Research in Human-centred Machine Learning (Publication IV)

It is possible to consider the overall machine learning pipeline of collecting data, building and inferring a model of the data, and making decisions with the model as an instance of human-AI collaboration. In publication II, we have presented an illustrative example of this setting, where an AI assistant was helping the human user build a linear model of their data as part of an exploratory data analysis task. We demonstrated that if the AI can infer the knowledge level of its user (e.g. knows collinearity or not), it can assist the user more effectively into choosing good models.

Throughout our work in publications I, II, and III, we have identified several challenges and important factors for modelling humans in human–AI collaboration, and their implications for human-centric machine learning. In publication IV, we bring together these insights, and argue that the human user must become an integral part of the machine learning pipeline. We imagine the future of building machine learning models as a human-AI collaboration task, where an AI assistant not only models the data but also the human modeller and the end user of the machine learning system. Then, the AI assistant can help the human modeller in building a good machine learning model for their data, that also satisfies the requirements of end users. Compared to post-hoc approaches to human-centric machine learning, this approach brings the human back into the modelling process from the very start, and is directly human-centred. However, realizing this vision requires significant advances in modelling human users, and also using such models interactively.

Publication IV presents our perspective on the core requirements of the needed advances in user modelling, and also the challenges ahead for developing them. It serves as a call-to-arms and a roadmap for further research in the direction specified above. By organizing existing knowledge, describing the core challenges, and providing preliminary directions of research for each challenge, it aims at bringing together the human–computer interaction and machine learning communities together towards the unified goal of developing effective AI assistants.
4. Concluding Remarks

The concept of an AI assistant is perhaps as old as the AI, however, so is inadequate assistance. Most of the readers who were fortunate (or unfortunate) enough to experience Microsoft’s Clippy (i.e. Microsoft Office Assistant) would know what inadequate AI assistance felt like. Systems such as recommender and information retrieval engines have come a long way since then, yet the way users are modelled has not. Modelling the users of an interactive intelligent system as passive sources of data could have been appropriate in the past, when the capabilities of the systems and user interactions were limited. However, the degree of interactions between intelligent systems and users has exploded recently, and the interaction has become so organic that humans started extending their theory of mind reasoning to AI systems. In the meantime, the capabilities of intelligent systems have grown rapidly, enabled by recent advances in machine learning. We believe that all these developments are culminating into a future where intelligent systems are considered as partners by humans, not as tools. This means the humans will see intelligent systems as agents, and thus these systems should see humans as such as well.

In light of the previous discussion, the overarching goal of this thesis has been developing multi-agent reinforcement learning methods to build better AI assistants, whose job is to augment human intelligence, not replace it. At the heart of our approach is the hypothesis that for effective assistance, a modern AI agent must model its human partners as agents as well. This distinguishes our approach from model-free multi-agent reinforcement learning which considers other agents as part of the environment, and models neither explicitly. From a technical standpoint, the model-based approach allows for sample-efficient inference when predicting future user behaviour from interaction data. In addition, from an ethical standpoint, the explicit modelling of human users means we can inspect the decisions of AI assistants based on their inferences about human users. For instance, we may find out that our AI calendar assistant is scheduling our agenda so that we perform certain tasks immediately, simply because it infers that we are likely to behave similarly to a time-
inconsistent agent (see section 3.2) and procrastinate.

In publication I, we demonstrated empirically that having a model of the strategic user behaviour improves the system’s performance in assisting the user, compared to treating it as non-strategic. We also presented the teacher–learner protocol as a model for the interaction between users and intelligent systems. Publication II modelled the cases where the user’s behaviour can in fact be harmful for assistance, if they lack the knowledge to interpret the AI’s assistance correctly. Then, the publication empirically demonstrated that, the AI assistant should try to teach the user in order to improve system performance in assisting the user. Publication II bridges an important gap between machine teaching and intelligent tutoring, by unifying how machine learning algorithms and human learners are modelled. This leads to a new research direction where machine teaching methods can take advantage of human pedagogical models. Publication III generalizes the scenarios of I and II into a setting where the user and the AI must together agree on what to do, by presenting the first formal mathematical model of centaurs (see section 3.2). It empirically demonstrates that, established models of human behaviour from cognitive science can be leveraged as inverse models, and then an AI agent can learn to assist its user most effectively. This is an important step towards developing online multi-agent reinforcement learning methods for human–AI collaboration scenarios where the AI must augment human intelligence. The intuitions gained throughout the development of publications I to III have culminated into publication IV, where we have identified core requirements and challenges for advancing user modelling, and presented them in an organized manner.

In summary, this thesis contributes to developing the online decision-making methods needed for AI assistants to help their human users. It achieves this by making advances in how the users are modelled, by developing multi-agent reinforcement learning methods that can make use of the user models both in inference and decision-making, and by identifying the future advances that are necessary.
References


