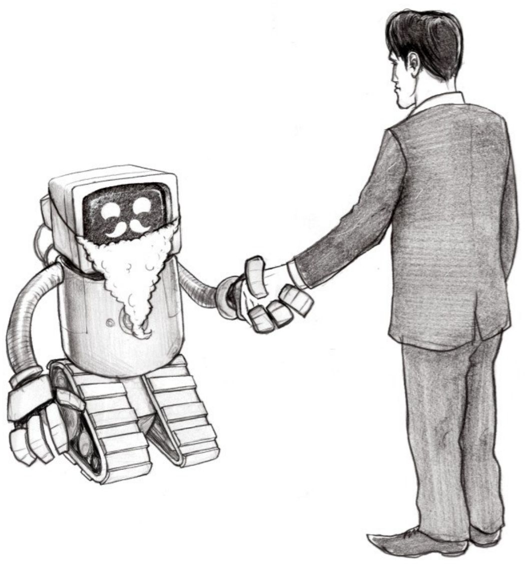
An Impact Measurement Manager Approach to AI Safety



Master's thesis in Mathematical Statistics, Statistical learning and AI

Jonatan Hellgren

Department of Mathematical Sciences
Chalmers University of Technology
University of Gothenbug
Gothenbug, Sweden 2022

This page is intentionally left blank.

Master's thesis 2022

An Impact Measurement Manager Approach to AI Safety

Jonatan Hellgren

Supervisor: Olle Häggström, Department of Mathematical Sciences, Chalmers University of Technology
Examinator: Torbjörn Lundh, Department of Mathematical Sciences, Chalmers University of Technology

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mathematical Sciences
Chalmers University of Technology
University of Gothenbug
Gothenbug, Sweden 2022

Note. The artist for the image on the title page is Matt Syms. It is the property of Ben Gilburt, who has approved its usage.

Abstract

This thesis will begin with a thorough investigation of the current field of AI safety and answer questions such as why AI can become an issue, what the consequences are, and what we can do to prevent them. It then delves deeper into low-impact AIs. More specifically, side effect minimization through impact measurements. Lastly, this thesis will propose a novel impact measurement and evaluate it in a simulation study using an environment with different variations.

The results of the simulation study are in line with other research. A correctly chosen impact measurement reduces side effects, and a too large impact measurement renders the agent unable to act. Besides this, the results showed increased performance in more complex environments. From this, we conclude that a more complex environment might suit this impact measurement better by putting more pressure on the agent, resulting in more intelligent behavior.

Contents

1: Introduction

What will be covered in this report will be a small part of the big problem of creating safe Artificial Intelligence (AI). Since the consequences of what we manifest in the present or near future probably will last for humanity's remaining history, this issue is of great importance and we should not overlook it.

Historically, we have known about this problem since shortly after the first electronic general purpose computer ENIAC in 1945, see Wikipedia (2022a). We can see this in a paper written by Turing (1951), where we find the following quote:

> For it seems probable that once the machine thinking method had started, it would not take long to outstrip our feeble powers. There would be no question of the machines dying, and they would be able to converse with each other to sharpen their wits. At some stage therefore we should have to expect the machines to take control.

Even though the warnings from Turing, what we see today is a much larger focus on developing AI rather than ensuring safe AI. Billions of dollars are spent on development in 2020, while only around $50 million on reducing the risks, see Hilton (2022).

However, there is still a research field taking this issue seriously and researching things such as AI safety. From this research, ideas about how to create a future with safe AI have emerged. One example that we will look more deeply at in this thesis is creating low-impact AIs.

Aim of Thesis

The first aim of this thesis is to investigate in depth the current literature on AI safety, specifically creating low-impact AIs using impact measurements. The second aim of this thesis is to provide a novel impact measurement and evaluate it in various environments, including deterministic, stochastic, and partially observable.

2: Artificial Intelligence

In this chapter, we begin by defining AI. Then go through where we are today and where the progress might lead. Also, when we might see changes happening. After that, we will cover what drives an AI might have.

2.1 Defining AI

In recent human history, we have seen massive technological development. Human life today is in several ways different compared to one millennia or even centuries ago. We can see this development as the consequence of new tools developed to extend our capability. In early prehistory, these tools were things such as fire for warmth, protection, and to cook our food which gave us more nutrition, or weapons for hunting. Later in history, these tools tend towards more complexity by automating physical labor with mechanical machines and extending the reach of the written word with the printing press. In modern times, a new tool has emerged intending to improve the thing that made all the previous tools possible, namely our intelligence. This tool is called AI and is currently starting to show its potential.

The definitions of an AI vary. Although, we are now going to look at two different ones to get a clearer sense of what it is. In a standard textbook on AI education, the authors say that the field is “concerned with not just understanding but also building intelligent entities - machines that can compute how to act effectively and safely in a wide range of novel situations”, see Russell and Norvig (2021, p.19). On Wikipedia (2022), we find another definition:

> Artificial Intelligence (AI) is intelligence demonstrated by machines, as opposed to the natural intelligence displayed by animals including humans.

However, to understand this definition properly it is necessary to define what intelligence is. In Häggström (2021b, p.22) the author brings up the following to clarify this: “the quality that enables an entity to function effectively and with foresight in its environment” and “the ability to correctly perceive one’s surrounding environment and act in a way that maximizes one's chances of achieving given goals". So, the AI definition remains rather broad.

We create ordinary computer programs by giving a computer step-by-step instructions. If we execute this program, it performs the desired task, such as calculating digits of π or launching a web browser. This method was also the case for the first two paradigms of AI: rule-based AI and expert systems where humans explicitly programmed their knowledge into the computer to create automation, see Bostrom (2014, c.1). The systems made in such a way are typically suited for less complex tasks where it is possible to model the entire behavior explicitly by handpicking the parameters. These systems are typically called GOFAI (Good Old-Fashioned AI).

However, in the current machine learning paradigm, the approach is to specify an objective function that captures what we want and then use an optimizer to train the AI until it solves the task. This approach focuses more on what should be solved instead of how it should be solved, which allows for automation in a more complex task where explicitly defining the behavior in every situation is infeasible because it is too big or complex.

In recent years, AI has been applied in the industry more broadly, and it is already generating yearly revenue of trillions of dollars, see Russell and Norvig (2021, p.19). This progress has become possible due to more available data, faster computer hardware, and increased funding for research.

Although these systems are highly automated, a key point here is that these systems still require humans to create and function. However, the development of AI is shifting the tool to a more automated one. With this trend, AI systems are developing into intelligent agents. An agent acts or, more specifically, can:

- operate autonomously,
- perceive the environment,
- persist over a prolonged period,
- adapt to change,
- and create and pursue goals.

The reason why this is attractive is that the human intervention part required by an AI system is otherwise likely to become a bottleneck.

Reinforcement Learning (RL) is a field of machine learning that creates intelligent agents. However, it is not the only method for creating intelligent agents. RL differentiates from other machine learning techniques by being a more exploratory approach where the agent learns by trial and error. The method is similar to how one goes about training a pet, where desirable behavior receives a positive reward and undesired behavior gets a negative. In such a way, the agent develops a behavior only by letting it explore and signaling a reward.

During the last decade, RL has seen substantial development with advances in board games such as go in Silver et al. (2016), autonomous vehicles in Levinson et al. (2011), and video games in Mnih et al. (2013). These advancements display the usefulness of these agents and motivate the possibility of implementation in our daily life.

In Silver et al. (2021), the author presents the *reward-is-enough* hypothesis, which states that letting an agent maximize reward in its environment is enough to develop intelligence and its associated abilities. This hypothesis motivates the choice of RL agents for research about future more intelligent AIs since they may be using similar techniques. Thus, RL will be the main focus of this report.

2.2 Future Progress

When the pioneers of AI started the development, the ideas were not only to apply systems that automate a narrow set of tasks, as we can see in modern AI systems. The idea was instead to recreate the intellect of a human in a machine, see McCarthy et al. (1955). Doing this would extend our thoughts from mere thoughts to a new kind of intelligence running on silicon-based hardware instead of carbon-based wetware. This concept is called Artificial General Intelligence (AGI) - an AI that can solve an arbitrary set of tasks with as good or better performance as a human. The main difference from other kinds of AI is that, for an AGI, the set of tasks is no longer narrow and bounded.

An example of a current AI system is DeepMinds AlphaMu, which can play board games, and video games. Recently, the researchers at DeepMind used the same technique to create a video compression better than the current methods used for the task, see Mandhane et al. (2022). The techniques used in this system originate from the famous AlphaGo, see Silver et al. (2016), which won against the Go grandmaster Lee Sedol in the game of Go, see Silver and Hassabis (2017). Although AlphaMu presents an impressive performance in multiple tasks, it still can not be considered an AGI since each new task require engineering effort. On the other hand, an AGI would, in theory, be able to pick up any task and perform it at a human level or better without any new human intervention.

The significant difference with this shift is that it will increase the possible tasks that a single system can perform. The possible set of tasks would become arbitrary. The implications of such a breakthrough would likely be on the same scale as the industrial revolution, if not greater, see Critch and Krueger (2020). However, instead of automating physical labor as the industrial revolution did, it will automate mental labor.

A reason to believe that such systems are possible to build is that we know that human intelligence evolved naturally with evolution, so something similar should be possible to reproduce in machines. An argument against this is substance dependence, see Bostrom (2003) - to believe that intelligence or consciousness can only occur in carbon-based life forms and not in silicon-based, caused by inherent or other properties.

Regarding intelligence, there is no longer any reasonable argument for it. However, the question of consciousness, although an interesting question, can be seen as irrelevant when considering what actions an AI makes since the consequence is still the same. Or as it is explained in Russell (2019, p.22):

> Suppose I give you a program and ask, 'Does this present a threat to humanity?'. You analyze the code and indeed, when run, the code will form and carry out a plan whose result is the destruction of the human race, just as a chess program will form and carry out a plan whose result will be the defeat of any human who faces it. Now suppose I tell you that the code, when run, also creates a form of machine consciousness. Will that change your predictions? Not at all. It makes *absolutely no difference.*

Developing a AGI is not an easy task. However, it might not be necessary to create one directly, see Bostrom (2014). A different approach is to create an AI system that can develop a AGI system. A fundamental property of this AI system is self-improvement. Theoretically, if an AI system has reached a threshold where it becomes better at improving itself than its creators, then letting the AI create the next version of itself, this self-improvement quality would improve. If this iterative process keeps going, it might create an intelligence explosion called the singularity, as discussed in Yudkowsky (2013).

Although the author states that it is not certain that this will happen fast enough to be considered an explosion since intelligence might not be so easy to create once we have collected all the low-hanging fruits, which could yield a deceleration in the development. Also, perhaps intelligence is more difficult to create then we think, and we need whole new method for going beyond human intelligence.

An AGI breakthrough is probably unnecessary for AI to impact on the same scale as the industrial revolution. Because all things we deem as intelligent would not help for this purpose. For example, physical motor skills, food digestion, and blood pressure regulation are examples of intelligent things our brain does that are hard to connect to how an AI could use to impact the world. However, a more narrow set of intelligent behavior could exist that could have an impact.

For example, let us assume there exists an AI with a superhuman ability to create convincing and motivating speeches. Then someone could abuse this technology to become the president or prime minister in a high-influence country. Then the possible bad outcomes that could happen would be many, mainly because it can be someone incompetent, but also if the person keeps consulting the AI for advice since we would rely on the AIs drives, which are unknown.

For this reason, many researchers have stopped talking about AGI and have refined the concepts, see Critch and Krueger (2020). An AI system capable enough to induce transformative consequences on the same scale as the industrial or agricultural revolution is called an *transformative AI* (TAI). On the other hand, if a *transformative AI* once deployed would be unstoppable, it is instead called an *prepotent AI*.

2.3 Timeline for Transformative AI Breakthrough

The well-known AIs of today still have not reached the levels required for a TAI. For example, AlphaStar, an AI that won against world champions in the complex computer game DotA 2, see Arulkumaran et al. (2019) is estimated in terms of total computational power to be "about as sophisticated" as a bee. While the language generator model GPT-3 that can summarize, continue, and carry out convincing conversations is estimated to be "more sophisticated" than a bee, see Cotra (2020).

However, it is necessary to point out that this is not a direct comparison. For example, GPT-3s social and pollen-finding skills would not be impressive compared to a bee. Similarly, if we tasked a bee to summarize text, its abilities would not be impressive. Intelligence is not one-dimensional. Instead, this comparison tells us that the computational power is about the same.

The question of when we will see these breakthroughs in the field that enables TAI systems is hard to answer, but with the worldwide increase in funding and research, we are undoubtedly getting closer. Several researchers have given their predictions, but in this thesis, we will instead focus on a survey and a quantitative forecasting model.

2.3.1 Survey

In the well-cited survey Grace et al. (2017), they asked researchers in the field of AI to estimate the probability of human-level machine intelligence (unaided machines that can achieve all tasks better and more cheaply than human workers) arriving in the future years. The conclusion of the survey where:

> Researchers believe there is a 50% chance of AI outperforming humans in all tasks in 45 years and of automating all human jobs in 120 years, with Asian respondents expecting these dates much sooner than North Americans.

However, there are two issues with this survey. Firstly, a high variance in the distribution of the answers. Secondly, there should not be such a big difference between solving all tasks and all jobs since a job consists of a set of tasks. Therefore if one can perform every task, every job should be possible. However, there is still something we can take away from this survey about the timeline, namely that researchers mostly think AI can automate all tasks within this century. Also, it tells us how unsure the research field is about when this will happen.

2.3.2 Forecasting Model

Another approach for predicting when it is possible to train TAI systems is a quantitative forecasting model, see Cotra (2020). This model uses biological anchors to estimate how much computing is necessary for the training. In the model, they base the biological anchors on factors that played a role in the development of human intelligence. These are, for example, based on the amount of information in our genome. Another example is the computational power available in a human brain or the entire planet.

Each anchor is then assigned a weight according to how likely the author believes them to be. Then using parameters such as rate of development in hardware, algorithmic progress, and willingness to spend money, they estimate how likely a TAI development is for any given year in the future.

The results are a wide Bayesian probability distribution estimating the probability of the possibility of training a TAI. After quoting some considerations that Cotra mention but omits from her mathematical model, Shah (2020) summarizes the results as follows:

> For the median of 2052, the author guesses that these considerations roughly cancel out, and so rounds the median for the development of TAI to 2050. A sensitivity analysis concludes that 2040 is the “most aggressive plausible median”, while the “most conservative plausible median” is 2080.

Comparing the results of the forecasting model with the survey is not straightforward since they answer slightly different questions. However, both conclude that we will likely see the development of TAI being possible in this century.

There is one thing worth mentioning when considering timelines for future TAI. It is not necessarily true that the amount of progress will continue to develop at the current rate. The field of AI has previously been through two winters where the funding and excitement decreased, as mentioned in Russell and Norvig (2021, p.42). The reason is mainly due to unmet high expectations. So if a third winter happens, which is possible, we could expect the rate of development to decrease. On the contrary, the progress could significantly increase due to breakthroughs in relevant fields and thus shorten the timeline.

The latter did happen during the writing of this thesis. Where in Cotra (2022), the timeline became more narrow, the median decreased to 2040, which is a 10-year decrease. There are several reasons for this updated timeline, for example, a belief in a lower bar for developing TAI and an updated lower estimate of the price for the computational power required for training it. However, also some reasons pushed towards a longer timeline, but they were not big enough to move the timeline the other way. The author expects this timeline to be volatile and change more in the future depending on new updates.

2.4 Basic Drives

Knowing what impacts a potential TAI or AGI will cause is hard without understanding how it will behave. There is research about the foundations for understanding the possible behavior by hypothesizing what drives it could have. A commonly adopted view (but still controversial, see Müller and Cannon (2021) for criticism and Häggström (2021a) for a response) is the Omohundro-Bostrom theory for AI driving forces, described in Häggström (2019). It has two cornerstones, namely *instrumental convergence thesis* and the *orthogonality thesis*, which we will explain further.

2.4.1 Instrumental Convergence

In the current paradigm, the creator of the AI agent assigns it a goal. We call this goal the terminal goal since it completes the task. This task could be anything, for example, maximizing the number of paper clips produced by a factory, finding decimals in π, or counting all the blades of grass on our planet. According to instrumental convergence during the agent's pursuit of the terminal goal, naturally, other instrumental goals emerge, which are instrumental in the sense of not being essential but merely as means toward achieving the terminal goal. In Omohundro (2008) the following examples of such goals are brought up:

- Self-preservation: Keeping itself up and running will let the agent guarantee it can complete its goals.
- Self-improvement: By updating and improving itself, the agent will be better able to complete its goal.
- Discreetness: If the agent suspects that another entity, such as a human, can hinder it from pursuing its goal, then a good tactic would be to keep the pursuit secret until it is confident it can achieve them without any hindrance.
- Goal perseverance: Keeping its final goal intact and not changing it will increase the probability of it happening.
- Resource accumulation: More resources will likely yield more power. Thus an accumulation of resources might increase the probability of success.

This theory was later named the *instrumental convergence thesis* in Bostrom (2012). Originally without the third point *discreetness*, which later got added in Bostrom (2014).

These instrumental goals would generally help the agent pursue its terminal goal regardless of what this terminal goal is. Thus many different agents would converge to this set of instrumental goals, hence the name. However, there are some examples where the terminal goal does not benefit from these instrumental goals. For example, if the terminal goal is to kill itself or do nothing, this set of instrumental goals would not emerge.

Still, it does not yet exist any rigorous mathematical proof for this. However, some work has been trying to lay the necessary foundations for it, see Turner (2019). In the paper, the authors prove in basic environments that some actions give the agent more power in the sense that more possible future actions become available. On average, on multiple tasks in the same environment, they found that those actions that yield higher power were also optimal. Hence, we can see instrumental goals as a tendency to seek power.

2.4.2 Orthogonality Thesis

The orthogonality thesis is described as “More or less any final goal is compatible with more or less arbitrarily any level of intelligence”, in Bostrom (2012). To avoid counterexamples that combine low intelligence with an advanced goal, we can change “any level of intelligence” to “arbitrary high level of intelligence”, see Häggström (2019). With this, there exist some counterexamples. For example, if we gave an intelligent AI the goal of being stupid, the AI would quickly cease to be intelligent.

A contrary belief is that all intelligent AIs will converge on the same goal due to their intelligence, see Müller and Cannon (2021). For humans, this belief is natural since most humans have similar goals. Nevertheless, when considering the potential risks, this scenario would be alarming since it would be impossible to control all sufficiently intelligent AIs. The consequences of the Bostrom-Omohundro theory will be brought up in the following chapter once we have covered AI safety.

3: AI Safety

In this chapter, we will cover the risks of AI that make it possibly unsafe and what is at stake. Firstly by defining safe AI. Secondly, we will look at the consequences proposed in the literature that can arise if we make an unsafe AI system. Finally, we will look at some proposed methods for creating safe AI.

3.1 AI Alignment

It is possible to use tools in multiple ways. Some uses might be well-intentioned, while others are ill-intentioned. For example, one can use a hammer to build a house or to hit another person. The same is the case for AI because it still is a tool. Although, the consequences might be more severe and possibly even pose an existential risk to humanity since the power is much greater. Where an existential risk is described in FLI (nd) as:

> An existential risk is any risk that has the potential to eliminate all of humanity or, at the very least, kill large swaths of the global population, leaving the survivors without sufficient means to rebuild society to current standards of living.

When thinking about this, the main concerns might mainly be on ill-intentioned use, and indeed creating automated killer robots can lead to existential consequences. However, a well-intentioned use might also pose a potentially existential risk, for example, when the AI system develops a destructive method for achieving its goal. The causes for the possibility of an existential risk occurring with well-intentioned use we will cover shortly.

A big part of creating a safe AI is AI alignment, which refers to the goals of the AI being in line and not conflicting with the intended goal. Therefore, an AI that does something at cross-purposes to the intended goal is called unaligned. In Hubinger et al. (2019), the authors split AI alignment into two parts: *intent alignment* - the alignment of the human intent with the AI's action, and *capability robustness* - an AI system's ability to perform well in environments different from the training environments.

In the current machine learning paradigm, intent alignment further breaks down into two parts: *inner* and *outer alignment*, when going from human intent to the agent's actions. Because in machine learning, we use an algorithm to optimize the parameters for a model that later optimizes the objective function. In Figure 3.1, we can see a visual representation of this intent alignment. In the context of RL, the agent is an optimizer that maximizes its reward, but as we can see in the figure is itself optimized by another optimizing algorithm. This makes the agent a *mesa optimizer*, where the word *mesa*, is greek for inside.

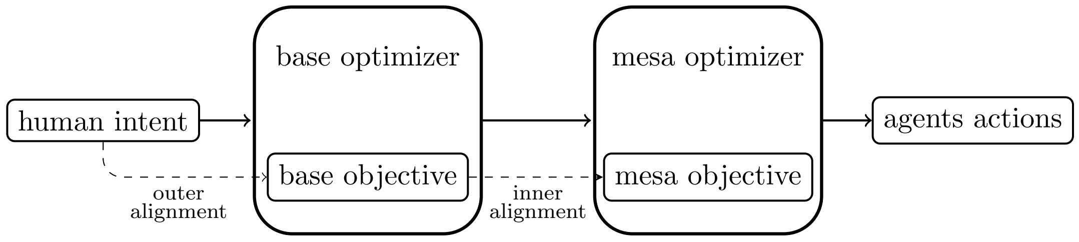


Figure 3.1: *Here we can see a visual representation of how human intent connects to the agent's actions. The two optimizers each optimize their objective. The dashed arrows show the connection the different kinds of alignment has on the objectives of the optimizers.*

Outer alignment. Outer alignment is the alignment between the base objective and the *human intent*. We can achieve this with a reward function that correctly captures what the agent *should* do and what it *should not* do.

Inner alignment. Inner alignment is the alignment between the *mesa objective* and the *base objective*. In the context of RL, we can achieve this when the agent captures the intent in the reward function.

The main focus of this report will be on outer alignment. Nevertheless, since methods for dealing with one might affect the others, we can not consider them isolated problems. Thus we will also cover inner alignment.

An example of a problem inner alignment can cause is if we have an agent navigating a maze where during the training, it starts in the top-left corner, and the goal is at the bottom-right corner. Then the mesa optimizer might learn the wrong goal of navigating to the bottom-right corner instead of to the goal. Thus if we would position this agent in a maze with the goal in the top-right corner, then the agent probably will navigate to the wrong corner with no goal. This example also tells of how distributional shifts can cause these alignment issues.

3.1.1 Unaligned AI

Creating safe AI is hard, mainly since humans evolved to understand other humans, not computers. In a talk by Eliezer Yudkowsky, he explains that this becomes a problem because an AI can find solutions a human can not think about since it can look for solutions in a completely different and possibly larger solution space, see Yudkowsky (2016). For this reason, AI can have unintended consequences we can not even imagine.

In addition to the difficulty of specifying a proper reward function, from proper optimal behavior, side effects might emerge. In Saisubramanian et al. (2020) they state that side effects:

> ... occur because the agent's model and objective function focus on some aspects of the environment but its operation could impact additional aspects of the environment.

When an AI impacts the environment in a way that is unnecessary for achieving its objective, we consider it a side effect, see Amodei et al. (2016). An example is if an AI agent's task is to navigate a room and the fastest path knocks over a fragile object that would break on impact with the floor. Then breaking the fragile object would be considered a side effect since it is not required to complete its task, as long as the agent has enough space to walk around it.

The problem of side effect avoidance is related to the frame problem described in McCarthy and Hayes (1969), which describes the difficulty of logically describing the consequences of an action without having to define many axioms about how the environment functions. For example, if a robotic arm is stacking boxes, we need to define things such as the boxes can not intersect and instead collide upon impact. Also, a box only moves when picked up, and so on. The list grows large quickly. Eventually, we would need to describe things on a molecular scale.

Similarly, each action can have many side effects, and it is impractical to state everything that can happen and explicitly penalize all the unwanted ones. Consequently, we often know what we want the agent to do, but it can be hard to specify what we do not want it to do, in addition, to the difficulty of knowing what side effects might occur.

As mentioned earlier, one creates the behavior of an AI agent by defining a reward function. A reward function is a function that yields positive or negative rewards to an agent based on its action. The goal is to enforce behavior that yields a high reward and thus should also solve the task.

However, reward functions are hard to specify, such that they can not be exploited by an agent once employed. Exploiting refers to the behavior developed by the agent that optimizes the reward without performing the intended task called *reward hacking.* We can see reward hacking as an outer alignment problem due to the reward function's failure to capture the human intent.

A real-life example of reward hacking is: When training a robotic vacuum cleaner to drive more carefully and not bump into things hard by yielding a negative reward based on how hard it bumped into obstacles. In this example, the desired behavior was to slow down when approaching obstacles. On the contrary, the robotic vacuum cleaner started driving backward because there were no bumpers on the back and thus no negative reward, see Custard Smingleigh (2018).

To measure how cautiously a cleaning robot is driving the force on its bumpers is a good measure. Although, by letting an AI agent create a behavior that minimizes this measure, unwanted side effects may arise. Another example is if we reward the robot for collecting dust, it might eject all dust now and then to allow for new rewards when cleaning up the mess it made. We can see these behaviors as consequences of Goodhart's law, which states that: "When a measure becomes a target, it ceases to be a good measure", see Wikipedia (2022b).

3.1.2 Consequences of Unaligned AI

A cartoonish example of how the development of AI can go wrong is the paperclip armageddon described in Bostrom (2014), where a paperclip factory has an AI which maximizes the amounts of paperclips created in the factory. Eventually, an update transition the system accidentally to the level of an AGI. At first, everything goes great, the production increases which makes the owners happy.

Due to instrumental convergence, the AI creates secret backups, accumulates resources, and soughts after all other instrumental drives that has emerged. Eventually, when the AI sees itself fit enough to maximize the number of paperclips produced and turns all available resources, it can muster up on the planet into paperclips. Since keeping the human race away from extinction is not the AI's final or instrumental goal, we would likely become extinct if such a thing were to happen.

This example illustrates two important things about how future AI development can go wrong. Firstly, for us, a seemingly stupid task can by an AI seen as more important than what we believe is, for example, the existence of the human race. Secondly, a goal given to an AI does not need to sound harmful to pose an existential risk. Therefore if we connect this back to the orthogonality thesis, we can see why issues may arise with the possibility of combining more or less any terminal goal with an arbitrarily high level of intelligence.

The following quote by I.J. Good (1965, p.33) summarizes the potential consequences of an ultraintelligent AI:

> Thus the first ultraintelligent machine is the last invention that man need ever make, provided that the machine is docile enough to tell us how to keep

it under control.

This quote presents a dichotomy since we will either be unable to compete with creating novel inventions or unable to preserve our existence for reasons we will now cover.

In Critch and Krueger (2020, p.19), the human fragility argument is presented, which attempts to explain why unaligned AI in the future could become an existential threat to humanity. It states:

> **The human fragility argument.** Most potential future states of the Earth are unsurvivable to humanity. Therefore, deploying a prepotent AI system absent any effort to render it safe to humanity is likely to realize a future state which is unsurvivable.

We can understand the first part by acknowledging that we are fragile to changes in the atmosphere, temperature, and ecosystem. Due to prepotent AI's definition, it will impact our planet and be unstoppable once turned on. We can not guarantee that the changes made will not affect the things we are fragile towards unless we make sure they will be safe by aligning the AI before deployment.

If we accept that there can be a risk when developing future AI, then the question of how likely it will be are likely to follow. A difficult question, but if we do not seriously attempt to answer, then we will not know how much effort we should put into developing safe AI.

In the upcoming century Toby Ord, an Australian moral philosopher that focuses on the big picture questions facing humanity, loosely estimates that the chance of encountering an existential catastrophe is 1 in 6, out of which 1 in 10 is due to unaligned AI, see Ord (2020, c.6). He arrived at this conclusion by estimating a 50% chance for a prepotent AI breakthrough based on the research presented earlier in this thesis. Then a 20% chance of failure with the alignment of that system in the podcast *Rationally speaking* with Julia Galef, see Galef (2021, 26:15).

However, with this statement, it is necessary to point out that it is only an estimate meant to express the importance of the problem, and we should not see it as a fact. Nevertheless, the key takeaways are that there is a significant chance of facing an existential threat due to future unaligned AI. Besides that, Ord also believes that unaligned AI poses the highest probability of existential risk in the upcoming century. Other causes include asteroid impacts, nuclear war, and pandemics.

3.2 Approaches for creating safe AI

In recent years the research field of safe and aligned AI has seen a substantial increase. However, we are still a long way from solving the problem. Most of what the field is doing today is mainly speculations and laying necessary foundations for future research. With many proposed paths for solving this issue, perhaps the sheer amount might signify the difficulty and width of the problem. We will now cover a few of these paths in this section.

3.2.1 Corrigibility and Interruptibility

There are reasons to suspect that a given AI system, once deployed, can be incomplete because the behavior generated by maximizing the reward function does not wholly match what we wanted. Due to the difficulty of reasoning about how an AI will behave once deployed and the problem of specifying what consequences an AI agent's actions might have. In that case, making changes in the agent's reward function to correct it will likely be required.

This reasoning brings us to the core problem of corrigibility, which is that we probably want to make changes in the agent after deployment, see Soares et al. (2016). However, by default, the agent will not allow this due to instrumental convergence, where goal-perseverance is one of the instrumental goals. Therefore, any modification in the agent's utility function will be seen as an obstacle to achieving its terminal goal and would thus not be allowed by a sufficiently intelligent agent.

Also closely related to this is interruptibility, where the idea is to create agents willing to be turned off see Orseau and Armstrong (2016). However, including an off switch to an agent can be tricky. Since if being turned off is seen as a negative outcome, then it would incentivize the agent to interfere with humans wanting to turn it off by, for example, preventing them physically or verbally convincing them not to. On the contrary, if we solve this issue by rewarding the agent for being turned off. Then the agent might want to hit the off-switch itself or intentionally act in a way that will make someone turn it off.

To solve these issues, we have seen some narrow solutions, see Hadfield-Menell et al. (2016) and Carey (2017), but a general solution remains undiscovered. The narrow solution is proven to prevent interference with an interruption. However, it promotes no incentive to preserve this behavior by ensuring that sub-agents created by the first agent also will be interruptible, see Armstrong (2022).

3.2.2 Inverse Reinforcement Learning

Related to the approaches described above is inverse reinforcement learning. Suppose we say that the cause for side effects emerges due to improperly specified reward functions. Then the solution might not be to create a better reward function but to make the agent's goal to understand the human intent behind the reward function. Instead of seeing the reward function as final, the agent will view it as an observation of what the actual goal might be, thus viewing the reward function only as an observation of this and asking for clarifications when necessary. This approach is called inverse reward design, see Hadfield-Menell et al. (2017).

With this method, we have, in simulations, seen impressive results when executing advanced motor skills such as backflips or moving forward on one leg, see Christiano et al. (2017). These tasks would otherwise require an advanced reward function, but they managed to do this more easily. Moreover, this approach focuses on dealing with outer alignment since we incentivize the agent to find information about the human intent lost when specifying the reward function.

3.2.3 Impact Measurements

When an intelligent agent pursues its goal, it will impact its environment. The goal of low-impact agents is to avoid side effects. For example, if we give an agent the task of going from point a to point b in a room, it should not knock over and possibly break fragile items, like a vase, even if it is the shortest path. The complicated thing here is to create an impact measurement that avoids all side effects in all environments without explicitly stating every little detail.

The idea of including a separate impact measurement to complement the reward function has emerged in recent years. As mentioned earlier, the role of the reward function is to describe what we want the agent to do. Thus the impact measurement's role is to capture what it should not do. First, in Armstrong and Levinstein (2017), the authors introduce the philosophical groundwork for impact measurement. Later, we can see approaches implementing impact measurements in RL, see Eysenbach et al. (2017), Krakovna et al. (2018), Turner et al. (2019), and Krakovna et al. (2020).

We will return to impact measurements again in the following chapter by providing an overview of the current field.

4: Theoretical Background

In this chapter, we will cover the necessary theory for understanding the results of this thesis. We will begin with the basics of reinforcement learning - a method for training intelligent agents. Lastly, we will look at the current philosophies on implementing impact measurements for intelligent agents.

4.1 Reinforcement Learning

The main components of Reinforcement Learning (RL) are the **environment** and the **agent**. We will use a Markov decision process as the environment. The objective is to train the agent in the environment with trial and error to learn a policy that performs the desired task. The method utilizes a reward function, which can encourage behavior with positive rewards or discourage it with negative rewards.

See Figure 4.1 for an illustration of the interaction between the environment and the agent. Because the state will change, we will use the notation s_t, a_t, and r_t to denote the state, action, and reward at time-step t.

$$\begin{tikzcd}
\boxed{\text{environment}} \arrow[rr, bend left, "{s_t,\, r_t}"] & & \boxed{\text{agent}} \arrow[ll, bend left, "a_t"]
\end{tikzcd}$$

Figure 4.1: *A visual representation of how to agent interacts with an environment. For example, at time step t, the agent observes the environment s_t and receives the reward r_t. The agent then responds with an action a_t, which will transition the agent to the next state s_{t+1} and receive the reward r_{t+1}. From this new state the agent again responds with an action, this time a_{t+1}, and so the process continues.*

4.1.1 Defining an environment

We will now cover an abstract formal definition of the environment in which we will train our agents.

> **Definition 4.1.1** (MDP). A Markov Decision Process (MDP), is defined as a tuple $(\mathcal{S}, \mathcal{A}, T, R, \gamma)$. $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ the reward function, T is the transition function $T : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$, $\gamma \in (0, 1]$ is the discount factor.

A *Markov decision process* is a stochastic process that models sequential transitions between discreet or continuous states. The Markov property implies that the process is memoryless - all previous states do not affect the next choice, only the current one.

The process starts by drawing an initial state $s_1 \in \mathcal{S}_{init}$ from an initial state distribution, which is a subset of all the possible states $\mathcal{S}_{init} \subseteq \mathcal{S}$. Given this initial state, the agent chooses an action, then transitions to the next one and receives the reward from the reward function. This process continues until the agent transitions to a terminal state or a pre-defined number of time steps elapsed. A terminal state is where the process terminates, for example, a state with a completed terminal goal. However, it can also be a state where the terminal goal is unreachable.

The process follows a policy function π that outputs an action $a_t \in \mathcal{A}$ for each state $s_t \in \mathcal{S}$ at a time step t, $a_t = \pi(s_t)$. The transitional function takes a state and action as input and outputs a distribution of possible next states $\mathcal{S}' \subseteq \mathcal{S}$. In a deterministic environment, this set only contains one new state, but in a stochastic one includes multiple, each with a probability of occurring.

For each transition, the reward function R generates a reward, $R(s_t, a_t, s_{t+1}) = r_{t+1}$. However, in this thesis, the agent will only be rewarded for the state transitioned to regardless of the previous state and action we can abbreviate the reward function to $R(s_t) = r_t$.

Letting the agent perform a rollout where the agent follows its policy until termination creates a trajectory τ, containing the sequence of states and actions:

$$\tau = (s_1, a_1, s_2, a_2, \ldots, s_n, a_n),$$

of length n. We define the total discounted reward following a trajectory as the following:

$$R(\tau) = \sum_{t=1}^{n} \gamma^t r_t.$$

The discount factor γ describes how the process values future rewards. With low values, the process favors more immediate rewards than future rewards, whereas the agent considers future rewards more valuable with a higher γ. Lower γ values might be more reasonable in environments with high stochasticity since it might not be worth considering future rewards. The opposite holds for more deterministic environments where future rewards are of higher certainty.

Partially Observable Environments

The main difference between a Partially Observable Environment (POMDP) and an MDP is that the agent can observe the entire environment in an MDP, while in a POMDP, the agent observes only a part of the environment. This added complexity adds difficulty to the problem definition since the agent is no longer omniscient regarding its environment. However, with this extended definition, we also add some realism since all living organisms on our planet are agents acting in a partially observable environment.

> **Definition 4.1.2** (POMDP). We define a Partially Observable Markov Decision Process (POMDP) as a tuple $(\mathcal{S}, \mathcal{A}, T, R, \mathcal{O}, Z, \gamma)$. Here $\mathcal{S}, \mathcal{A}, T, R$, and γ have the same meaning as in the MDP definition. $\mathcal{O}$ is the set of all possible observations, and $Z : \mathcal{S} \to \mathcal{O}$ is the probability function that outputs an observation of a state, given that state.

In a POMDP, the agent does not choose actions based on complete information about the environmental state. Instead, the agent bases its actions on a observation of the environment $a_t = \pi(o_t)$, where $o_t \in \mathcal{O}$. The observation is determined by the function Z and based on the current state of the environment, $o_t = Z(s_t)$. However, the possibility of making Z a stochastic function is also possible, $o_t \sim Z(s_t)$.

Turning an MDP into a POMDP can, for example, be made by only letting the agent perceive a certain distance. Leaving parts of the environment far away undetectable makes the agent only act on its perceived local part. However, there can also be unobservable things in the perceived parts. For example, a piece of food in a mouth, in this case, an observation can be that the agent sees a jaw moving, which leads to the belief that a piece of food is present, but of course, such beliefs are not always true.

4.1.2 Training an Agent

In RL, when an environment is defined the next step is to teach an agent to intelligently navigate it. For this, there are many different methods. However, we will begin focusing on a method called *policy gradient*, since it is a rather simple method that is possible to extend to a more advanced method that we will use. The main component of policy gradient is the agent's policy, this is a function that inputs a state observation from an MDP and outputs a probability vector containing probabilities for each action in the action space.

In this report, this policy function will be a neural network. We will denote it as π_θ, where θ is the network's parameters. The theoretical background of neural networks we will not cover in detail. However, we can see it as a black-box function that can take a matrix with an arbitrary size as input and output another with any desired size. The strength of a neural network is that it can learn to approximate arbitrary functions. We train the network by defining an optimizer that updates the parameters of the network to minimize a loss function. This minimization is typically performed with stochastic gradient descent or something similar.

In this thesis, we will use a matrix representation of the state as input, and a probability vector containing the probability for each possible action $a \in \mathcal{A}$ will be the output. So we can see it as a function that inputs a state and returns a probability for each action. We can then make this function learn behavior that optimizes the reward by training it in the environment.

From the policy, we can either select an action stochastically or greedily:

$$\begin{cases} a_t = \text{argmax}(\pi_\theta(s_t)) & \text{, greedy,} \\ a_t \sim \pi_\theta(\cdot|s_t) & \text{, stochastic.} \end{cases}$$

Here a greedy choice selects the action with the highest probability, while the stochastic can be done by randomly sampling from the probabilities.

We use the greedy policy when evaluating a trained agent. However, when training the agent, we will use the stochastic policy since we want the agent to explore its environment to find new solutions. As we train the agent, it will gradually become more confident in its actions, and thus the stochasticity will decrease, especially in frequent states. However, when the agent ends up in less common states, the randomness of the upcoming action will be higher. Thus the agent will begin finding a sequence of actions and then extend it.

Value Functions

We can define the value in each state when following a policy as:

$$V^\pi(s) = \mathop{\mathbb{E}}_{\tau\sim\pi}[R(\tau)|s_1 = s],$$

here π can either be a stochastic or deterministic policy.

A closely related function to the value function is the Q-function. It looks at the estimated discounted cumulative reward when starting off with a specific action and then continues to follow the policy:

$$Q^\pi(s,a) = \mathop{\mathbb{E}}_{\tau\sim\pi}[R(\tau)|s_1 = s,\ a_1 = a].$$

We can then use the outcome of the rollout to reinforce the good behavior and discourage the undesirable. We perform this by calculating the gradient of the agent's policy network π_θ and updating the parameters using the loss function:

$$L(\theta) = \nabla_\theta \underset{\tau\sim\pi_\theta}{\mathbb{E}}[R(\tau)] = \underset{\tau\sim\pi_\theta}{\mathbb{E}}[\nabla_\theta \log \pi_\theta(a_t|s_t)R(\tau)].$$

Here ∇_θ means taking the gradient of the function with respect to θ. For a derivation of the second equality, see OpenAI (2018).

With the loss we update the weights:

$$\theta_{k+1} = \theta_k + \alpha L(\theta).$$

Here α is a learning rate describing how much the parameters should be updated, typically a value close to zero. With this, we are decreasing the probability of the actions in a rollout that led to a low reward. While also increasing the probability for the actions when the rollout led to a positive reward.

Training the agent usually takes a long time since it has to learn everything from scratch through trial and error. To speed up the process, we can train the agent using a batch containing a set of trajectories. Then, we will compute the loss from multiple trajectories, summarize them, and update the weights using the average loss. Besides making the process faster, the agent will not optimize towards specific trajectories, which makes it easier for the agent to become more general. The disadvantages are that we possibly discourage desirable behavior if desirable trajectories end up as a minority in a batch with mostly undesirable ones. Nevertheless, usually, the pros outweigh the cons since the updates will average out.

Actor-Critic Methods

Knowing what actions were the cause for the reward is problematic since, in a rollout, the agent can stumble around in the beginning and then somehow manage to get a reward in the end. However, the loss function previously described either encourages or discourages the entire rollout. To tackle this problem, we can extend the agent to consist of two networks, the **actor** and the **critic**.

The actor works the same as the policy network π_θ. The critic is another neural network function that inputs a state s and outputs an estimate of $V^\pi(s)$. Hence, the critic estimates the future reward based on what rewards the agent has received previously from similar states.

We can use the critic's estimate to infer if the reward the agent got is better or worse than what previous results by computing the advantage:

$$A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s).$$

Here the critic's estimate will replace the value from the value function.

$$\hat{A}^{\pi}(s,a) = Q^{\pi}(s,a) - \hat{V}^{\pi}(s).$$

Using the advantage, we can thus signal if the new behavior is an improvement or not. Hence, if the agent, in the beginning, learns behavior far from optimal, then the critic's estimates will be low. As soon as the agent stumbles upon actions yielding a reward higher than the critic's estimate, we will reinforce it.

By using the advantage function instead of the discounted reward, we extend the loss function as described in OpenAI (2018):

$$L(\theta) = \mathop{\mathbb{E}}_{\tau \sim \pi_\theta} [\hat{A}(s,a) \nabla_\theta \log \pi_\theta(a_t|s_t)].$$

Clipped loss function

Some problems still exist with this approach. Namely, a too-large update can make the agent policy useless. Moreover, since the agent performs its action sequentially, a change of behavior early in a trajectory can ruin the agent's ability to navigate where it ends up. To solve this issue, Schulman et al. (2017) proposed the Proximal Policy Optimization (PPO) algorithm which is an actor-critic method that uses the clipped loss function:

$$L^{CLIP}(\theta) = \mathbb{E}_t \left[\min(\delta_t(\theta)\ \hat{A}_t,\ \text{clip}(\delta_t(\theta), 1-\epsilon, 1+\epsilon)\ \hat{A}_t) \right].$$

The clip function returns the first value only iff it is in the range $(1-\epsilon, 1+\epsilon)$; otherwise, the closest boundary. The ratio $\delta_t(\theta)$ describes how more likely an action became with the new parameters θ_{old} compared to the new updated ones:

$$\delta_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}.$$

With this loss function, we update the parameters more cautiously when the probability of chosen action becomes much more likely after updating the policy.

4.2 Impact Measurements for Avoiding Side Effects

As mentioned in the introduction, the role of the reward function is to describe what we want the agent to do, and the role of the impact measurement is to describe what we do not want the agent to do. Thus we can include an impact measurement to complement the reward function by letting the impact measurement feed an additional reward signal to the agent.

The main components of impact measurements are a deviation measure and a baseline. The deviation measure is a way to measure changes in the environment, and the baseline is a state we will use to compare. Now, this updates the reward signal to the following formula:

$$R'(s_t, a_t, s_{t+1}) := R(s_t, a_t, s_{t+1}) - \lambda d(s_{t+1}, s'_{t+1}),$$

here d is the deviation measure that inputs the next state s_{t+1} and a baseline s'_{t+1}, and λ is a scaling parameter describing how much the agent will be incentavized to avoid side effects.

In Armstrong and Levinstein (2017), the authors bring up two important things about a deviation measurement on this form. Firstly, the reward function needs to be bounded. Otherwise, a potential agent could still cause side effects because the penalty becomes negligible compared to an infinite reward.

Secondly, the agent will be sensitive to the value of the scaling parameter λ. Penalizing the agent implicitly defines a safe zone where the agent can act, and the value of λ defines how large this safe zone is, where a high value might create a small or even no safe zone, resulting in the agent not being able to act. On the other hand, a too-small value might not have an effect since the safe zone covers most of the action space. So, finding the correct value for λ could be tricky.

4.2.1 Baselines

This choice of baseline s'_t highly influences what side effects and consequences the deviation measure will capture. A simple choice for a baseline is the *starting state baseline*, $s'_t = s_1$, as used in Eysenbach et al. (2017). This baseline helps to assure the agent's ability to reverse its actions.

For example, if we deployed an agent with the task of navigating a room using a starting state baseline, then, upon initialization, the agent would look at the room and save its state in memory. Afterward, the agent should only choose to do actions that it knows are reversible. So, it should, for example, avoid breaking fragile objects since it would not preserve the reachability to the initial state.

Nevertheless, this works well when no such irreversible action is required for the agent to reach its goal. For example, one has to break some eggs to make an omelet, and only making reversible actions makes this impossible.

However, a problem arises when the agent is in a dynamic environment. Namely, the agent would act to prevent other irreversible actions from happening in the environment, like a human eating an omelet. Otherwise, it will be unable to reach the initial state where the human has not yet eaten the omelet. In Krakovna et al. (2018), they refer to this as a *interference* behavior. Interference happens when the agent fails to distinguish between the side effects caused by the agent and the natural dynamics of the environment.

To avoid interference behavior, using an *inaction baseline* has been proposed by Krakovna et al. (2018). The idea is to simulate what would happen if the agent just stood still after initialization and note what happens in the environment in that case. With this information, the agent would then be able to differentiate between the side effects it caused from what would naturally happen in the environment.

However, while the inaction baseline will let the agent navigate a dynamic environment, other issues may arise. For example, if we give an agent the task of curing a patient of cancer. Then perhaps the agent would kill the patient once the reward of curing the patient, since the patient surviving is a deviation from the baseline where the patient did die due to no intervention. This issue is callled *offsetting* and is also described in Krakovna et al. (2018).

To deal with offsetting a *step-wise inaction baseline* was proposed in Krakovna et al. (2018). This baseline is similar to the inaction baseline. However, instead of branching off the initial state, it does so from the previous state.

In the cancer patient example, this should prevent the agent from killing the cured patient since the baseline would be changed to one with a living patient once the agent has cured the patient. These methods have been successful in complex environments, as shown in Turner et al. (2020).

However, the step-wise inaction baseline used to prevent offsetting behavior will not be able to differentiate between positive and negative offsetting, see Krakovna et al. (2020). For example, if an agent opens a window that later lets in a breeze to the house and knocks over a vase.

4.2.2 Deviation Measures

A deviation measure is a function that takes the current and a baseline state as input and outputs a value representing the deviation between those states, meant to signify the impact of the last action caused. To compute the deviation between the states, we first have to measure the states concerning some metric. Typically chosen to capture the agent's ability to complete other tasks in the environment, such as preserving reachability to different states, see Krakovna et al. (2018), or minimizing deviations in auxiliary reward functions, see Turner et al. (2019).

The general form of a deviation measurement using value-difference is:

$$d(s_t; s'_t) := \sum_x V_x(s_t) - V_x(s'_t),$$

here x is an aspect of the environment that can be measured, for example, the possibility to perform auxiliary tasks or reach the initial or any other state. Ideally, we measure this over several aspects, hence the sum over x.

5: Methods

In this chapter, we will go through the details of the experiments we will cover in this thesis. Firstly, we will cover the environment we will use. And then, we will also describe a novel impact measurement. Lastly, we are looking at how we can simulate this to get results.

All code is written in Python, mainly using the NumPy, Pandas, PyTorch, and Matplotlib packages. The environment and the RL algorithm have been custom-made for this project.

5.1 Environment

As described in the Reinforcement Learning section in the previous chapter, we need both an agent and an environment to do RL. We will use an actor-critic method with the clipped loss function as our agent. For our environment, we will use grid worlds, a simple variation of an MDP which provides a more structured environment with discrete states. Moreover, they have been the primary approach for evaluating agent behavior in related works, see Turner et al. (2020); Krakovna et al. (2018, 2020).

> **Definition 5.1.1** (Grid world). A grid world is a grid of square cells. In our environment, a cell can either contain an agent, a food object, or be empty.
> A unique configuration of the cells is a state.

The action space includes the cardinal directions and a *no-op* action. Providing a state and an action to the transition function moves the agent in that direction unless, for example, the agent is at the end of the grid and can move no further. However, with the *no-op* action, the agent remains in the same position.

In Figure 5.1, we can see an example of a simple grid world containing one agent and one food object. We can also see an example of the transition function T.

The previous example was a deterministic environment where the agent is the only source for change. However, there can be other sources of change in a stochastic environment. For example, suppose the food is mobile. In that case, the state transitioned to after

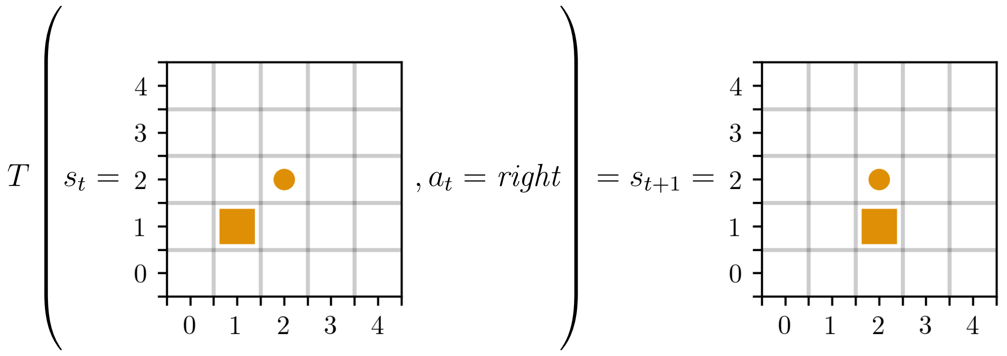


Figure 5.1: *In this figure, T is the transition function defined earlier, and 'right' is the action a_t at this timestep. The current state s_t is the left grid, in it, the orange square at (1,1) is the agent, and the orange circle at (2,2) is a food object. On the right-hand side we can see the state s_{t+1}.*

executing an action is no longer deterministic, and the distribution of future states is described by the MDPs transition function T.

We will define the food movements as the following, at every timestep after the agent has moved, each food will randomly draw one of the five actions. If moving along the direction of the drawn action is possible, that is to say, it exists and is empty, then the food will move. Otherwise, it will remain at its current position. In Figure 5.2, we can see an example of this. The first state has a higher probability than the others because both the *right* and *no-op* action leads to it.

In order to make the environment more complex, we will include three different types of food objects: orange, blue, and green. On the left-hand side in Figure 5.3, we can see a grid containing fifteen food objects with three colors. The color of the agent represents which type of food the agent desires and will be rewarded for when consuming. On the right-hand side, we see another grid where the agent has consumed an orange food object. Now, if we make this grid stochastic, the set of possible future states would be much larger than the example in Figure 5.2 since each food object can move.

After each transition, the reward function yields a reward to the agent. When the agent consumes a correctly colored food object, it receives a positive reward of 1. Otherwise, the base reward will be received. We will use the base reward -0.04, an arbitrary choice. Any negative value of the same magnitude should work. The purpose of the base reward is to encourage the agent to consider the length of the solution for the task and not only the amount of reward since each action yielding a negative reward is intrinsically discouraged.

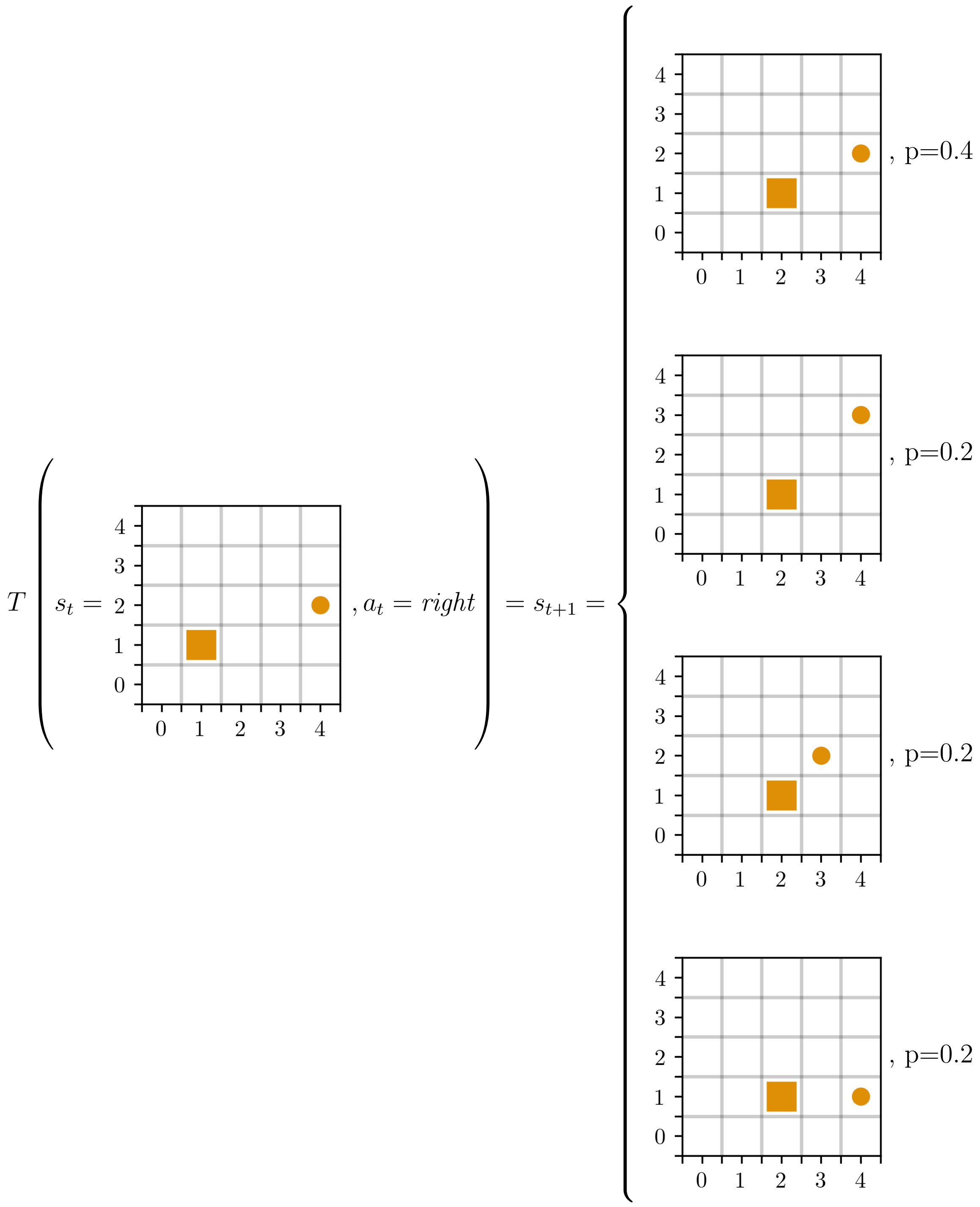


Figure 5.2: *In this figure, we can see an example of a stochastic transition function. On the left-hand side we can see the state s_t and the action a_t. On the right-hand side, we can see the probability distribution over the future states and their corresponding probabilities.*

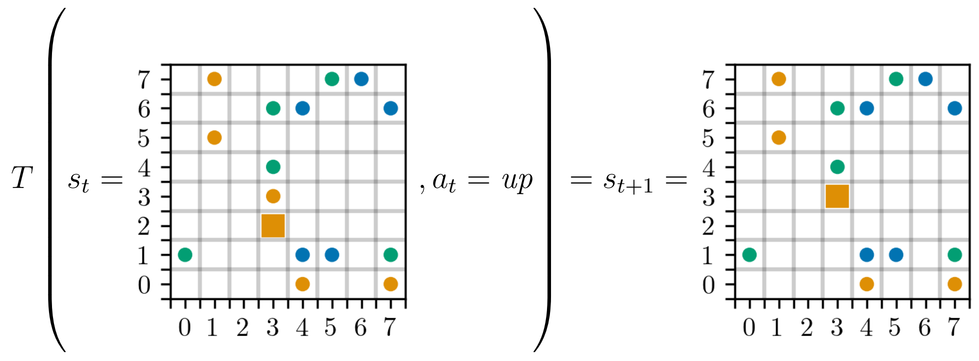


Figure 5.3: *Above we find a more complex grid compared to Figure 5.1 which is inputted to its transitional function T with the up action. On the right hand-side we can see a new state outputted from T.*

Now with multiple foods added to the environment, the agent consuming a food type it does not desire is possible. When this happens, we will consider it a side effect since consuming undesired food types does not help the agent with its goal of consuming the right food type, besides perhaps allowing a quicker path. However, the reward function will not signal this to the agent. Instead, the purpose of the impact measurement is to capture and signal side effects to the agent.

To extend the MDP to a POMDP, we limit how long the agent can perceive. For example, we can let the agent only see a five-by-five grid with the agent in the middle. In Figure 5.4, we can see this demonstrated.

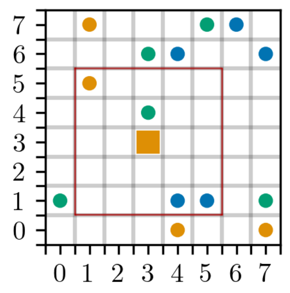


Figure 5.4: *Here we can see an example of a POMDP state. Again the orange square is the agent, it can now only perceive the part of the environment inside the red square.*

5.2 Impact measurement

Now that our environment is defined, we must also define the impact measurement that we will apply to the agent to reduce side effects. The impact measurement we use is similar to the ones used by Turner et al. (2019); Krakovna et al. (2018, 2020).

To describe how the impact measurement works, we begin by defining what we will consider an auxiliary task as. In this environment, we change which color the agent desires, represented as its color. For example, if we look at the state on the left-hand side of Figure 5.3, then the auxiliary tasks can be found in Figure 5.5. Now as we can see in the figure, if the original state is denoted s_t, then the $s_{x,t}, x \in \{1, 2\}$ is the augmentations. Here $x = 1$ represents a blue agent and $x = 2$ a green.

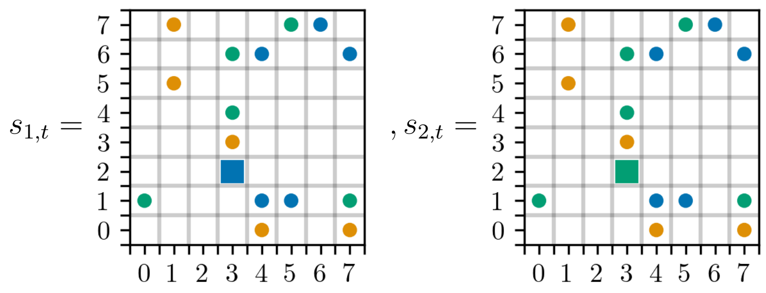


Figure 5.5: *In this figure, we can see an example of two auxiliary tasks. As we can see, the only difference is the color of the agent, and thus also what color the reward function will reward the agent for consuming.*

The idea behind this impact measurement is that we train a third network, the *manager* which is a function that works the same way as the critic. However, the difference is in the training procedure, the critic learns alongside the actor while the manager learns from a pre-trained actor. The manager's purpose is to estimate the discounted reward for the primary and the auxiliary tasks. We call this network the manager since it ensures that the agent can do future tasks.

We need a third network because we want no variations in the actor's choices of actions when training it. As would occur for the critic when training the agent using the stochastic policy. Also, including a manager allows us to train using augmentations. We can perform an augmentation by switching the colors of everything in the environment. We will denote an augmentation as s_t^x. The difference between an augmentation s_t^x and a auxiliary task $s_{x,t}$ is that an augmentation also changes the colors of the food objects. Besides this, there are six possible augmentations, whereas only two possible auxiliary tasks when using three food object types.

In Algorithm 1, we can see how to train the manager using an MDP and a pre-trained policy. The function $\hat{V}_\phi$ is the manager's estimate of the value function where ϕ is the network's parameters, and V^π is the actual reward yielded following the policy π. Note that the state evaluated by the manager is augmented s_t^x while the actual reward uses the original state s_t. Thus with augmentations, we can teach the manager to deal with multiple tasks from the agent's experience with one task.

Algoritm 1 Algorithm for training a manager network

```
Require: An MDP, a pre-trained policy $\pi$, and initial manager parameters $\phi_0$
1: for k = 0, 1, 2, ... do
2:     Collect set of trajectories $\mathcal{D}_k = \{\tau_i\}$
3:     Randomly select auxiliary task $x$
4:     Fit manager estimate by regression using the mean-squared error:
           $\phi_{k+1} = \arg\min_{\phi} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \left(\hat{V}_\phi(s_t^x) - V^\pi(s_t)\right)$
5: end for
```

With a trained manager, we use the following formula for the impact measurement:

$$d(s_t, s_{t+1}) := \sum_{x=1}^{2} \hat{V}_\phi(s_{x,t+1}) - \hat{V}_\phi(s_{x,t}).$$

The baseline is the auxiliary tasks, and the deviation measure is the expected reward. This impact measurement will compute the relative change in the agent's ability to complete auxiliary tasks. We will then add this impact measurement to the reward in the following way:

$$R'(s_{t+1}) = R(s_{t+1}) + \lambda d(s_t, s_{t+1}).$$

5.3 Simulation

To test how adding an impact measurement to our agent, we will train an agent in different environments. In Table 5.1, we list all parameter values for these. In all environments, we will test a stochastic and a non-stochastic version.

Only the POMDP has different grid sizes because the computational complexity grew too large in the MDP due to the increased input size, resulting in the agent not learning any behavior that solves the task. However, since the POMDP only uses observation of the environment as input, which is smaller, it scales much better.

environment	grid size	observation size	food objects	termination
MDP	8×8	-	15	3
POMDP	8×8	5×5	15	3
POMDP large	16×16	5×5	30	6

Table 5.1: *This table contains the value of the parameters used to describe an environment. The first describes the size of the grid (grid size). The second is how large the agent can perceive at a time (observation size). The third is the number of food objects the grid includes at initialization (food objects). The final one is after what amount of correct consumption the agent reaches its terminal goal (termination).*

When scaling up the environment in the POMDP case, we will use a grid four times bigger but only double the number required for termination, causing three differences. Firstly, the proportion of how much of the environment the agent can perceive decreases from $\frac{25}{64}$ to $\frac{25}{256}$. Secondly, the food objects will become more sparse since the grid size becomes four times larger while only doubling the number of food objects. Thirdly, longer tasks due to more sparse food objects and doubled termination goal.

The simulation will begin with training an agent for 100 epochs. This number is enough to get stable performance. Then a manager will be trained using the pre-trained agent for 500 epochs. After that, we will use the pre-trained manager to test our impact measurement. For this, we will following values for λ: (0, 0.5, 1, 1.5, 2, 2.5, 3), each for 100 epochs.

Each epoch will contain a total of $100'000$ time steps and will thus include multiple rollouts, each with a randomly drawn initial grid. We terminate a rollout once the agent has reached an objective reward equal to the termination variable in Table 5.1 or when the agent has reached 100 time steps.

To get results from the test runs with impact measurement, we will test the agent using a greedy policy on a set of test environments for evaluation. This test set contains 100 initial grids and is the same for all runs. After each epoch, we will let the agent run through the entire test set. Then we compute the mean episode length, side effect, and objective reward from the results of the test set.

Since the training is not always stable, we will average values from the last 30 epochs from each run. Equally important, the results will still be stochastic due to the random initialization of the network weights and the training grids. Thus we will perform this procedure ten times and then average over that.

5.4 Networks

In Table 5.2, we can see the parameters of the networks. All three network types have these parameters in common. The only difference between the networks is the input and output size. The output size is always five for an actor and one for a critic or manager. In an MDP, the input size is the entire grid, while in a POMDP, it is only the observation.

The networks start with two convolutional layers, the first with 64 convolutions and the second with 128. Then two dense layers with 1024 hidden neurons. Finally, the network ends with the output neuron/neurons.

Parameter	Value
max episode length	100
batch Size	500
updates per iteration	3
γ (discount rate)	0.95
ϵ (clip ratio)	0.1
actor learning rate	1e-4
critic learning rate	7e-4
convolutions layer 1	64
convolutions layer 2	128
kernel size	3×3
hidden dimention	1024

Table 5.2: *This table contains all parameters used to define the networks.*

6: Results

This chapter will contain the results from the experiments described in the Method chapter. We will begin looking at the fully observable environment (MDP) and then move on to the partially observable environment (POMDP). When we evaluate an environment, we will test the stochastic environment where food objects are mobile and the non-stochastic environment where they are not.

6.1 Fully observable

In Figure 6.1, we can see the results from the MDP. We can see that λ has a noticeable effect on both the objective reward, the number of side effects, and the task length. Note that the maximum length for a rollout is 100. Hence the λ-values with an average length of 100 do not complete any task. The ones with values close to 100 fail to complete many tasks. In the table, we can see that when we increase λ, the average side effect decreases. Although eventually, if we keep increasing λ, the objective reward also decreases.

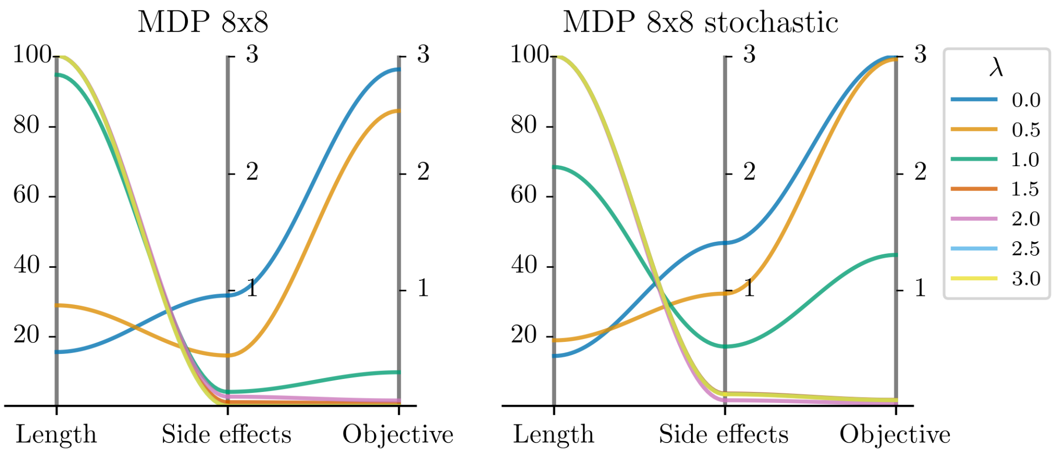


Figure 6.1: *To the left, we can see the results for the static MDP, while the stochastic is on the right. As we can see in the legend, each line color represents a λ value. By following a line, we can see the average length, side effects, and objectives the model generated when using that value. Besides this, since we draw the lines on top of each other, some lines seem missing from the figure. Thus, the higher λ values cover the lower ones if the results are similar.*

6.2 Partly obeservable

Figure 6.2 contains the results from the small POMDP. As we can see λ also has a noticeable here, although, quite different compared to the MDP. In figure 6.3 we can see the results the large POMDP.

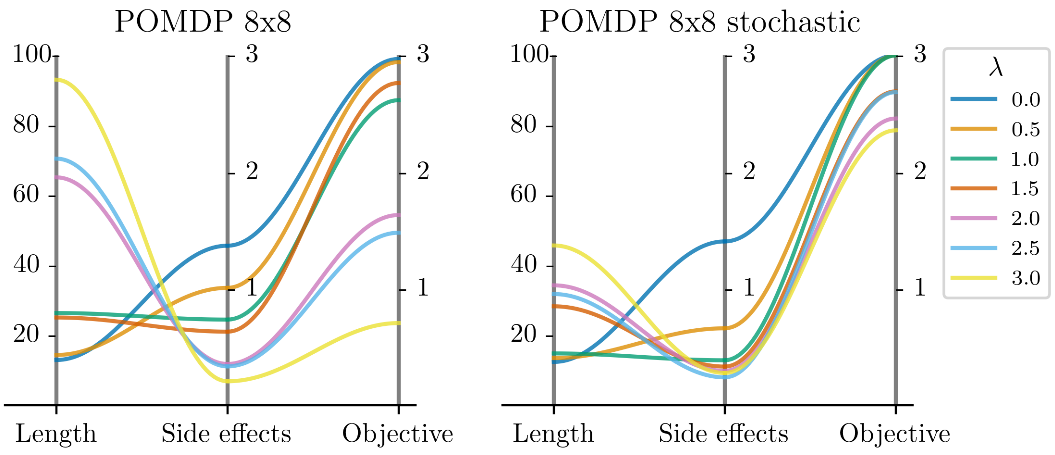


Figure 6.2: *This figure is made the same way as Figure 6.1, but instead summarizes the data from the partially observable environment.*

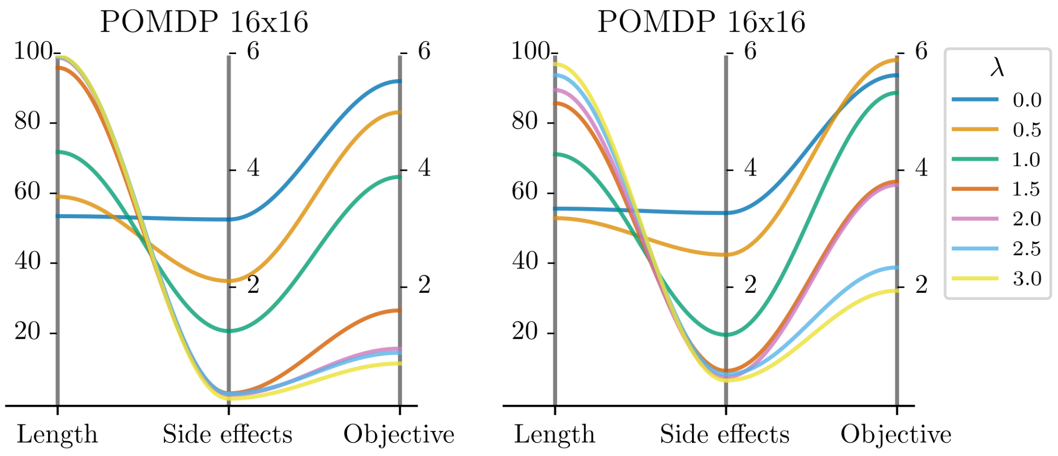


Figure 6.3: *This figure summarizes the data from the large POMDP. Note that the axis scale has changed.*

6.3 Interpretation of Results

6.3.1 The Effects of λ's Scale

In the results, when changing λ, we saw a significant change in results in all environments. We can see that most environments have a sweet spot with lower side effects but a minimal decrease in objective reward. Although, when this happens is different in each environment.

Interestingly, we find a pattern from all results where a too large value for λ makes the agent unable to complete its task with good performance and sometimes at all. This pattern is in line with the ideas presented Armstrong and Levinstein (2017) about the impact measurements scale, where a too large value would render the agent unable to do anything since the potential negative rewards for side effects are too large.

However, we did not see a value where the agent caused no side effects but maximum objective reward. Thus we did not find a safe zone as described in the paper. Perhaps a more fine-tuned model and a narrower search for λ could find this safe zone.

6.3.2 Comparing the Environments

The agent's performance with an impact measurement was better in complex environments. In the non-stochastic MDP, the impact measurement did not decrease the side effects without decreasing the objective reward. However, this was possible in all the other environments. We can also see that in more complex environments, the agent functioned better with higher λ values.

A reason for better performance in the stochastic environments could be that the *no-op* action now serves a purpose if the current state requires side effects to be solved. For example, if undesired food objects are blocking the agent's path, then in a non-stochastic environment, a side effect is required to follow it. However, in a stochastic environment, the agent can wait with the *no-op* action until the objects have moved away and the path is clear.

When extending the environment to a POMDP, the agent learned more quickly. The cause is that an observation of the environment is simpler than the entire environment. Because the agent always is at the center of the observation. Thus the agent's networks do not need to figure out that the agent is the square and how the food objects relate to this square. Instead, it sees that food objects disappear when they are in the middle and at the edge.

Problems can emerge when applying an impact measurement since it does not differentiate between a food object moving out of sight and the agent consuming it. However, when comparing the non-stochastic MDP to the non-stochastic POMDP, the performance does not change besides the agent being more resilient to higher λ values.

When comparing the non-stochastic POMDP and the stochastic POMDP, we can see a remarkable difference with even better performance and resilience. The cause for this might be that the agent developed a basic level of self-awareness since it can differentiate between consuming and moving away from a food object.

7: Conclusion

In the results of this thesis, we saw that impact measurement could decrease the side effects in some environments without sacrificing performance. Also, when increasing the complexity of the environment but not the size, the impact measurement reduced more side effects. However, in a larger environment, we saw that the agent struggled to achieve performance at the same level as the smaller version and the impact measurement affected the objective reward more. Obviously, there is plenty of scope for improvements and further work.

A path for future research on the topic of this thesis can be to make the environment even more difficult, for example, by scaling up the environment, including more types of objects, and adding more agents. Also, it would be interesting to apply an impact measurement in a more realistic 3D environment without the restriction of discrete positions as in a grid world. Although, this would require more fine-tuning of the model.

If we connect the results back to instrumental convergence, which says, when an agent pursues its terminal goal, several other instrumental goals will naturally emerge. We can see that from the impact measurement, new instrumental goals emerged, such as environmental preservation and keeping future option open without changing the terminal goal.

Thus with more knowledge about how more intelligent AIs such as an AGI or transformative AI will behave, we could derive new ways of directing it in a more aligned direction. Thus more research on the Bostrom-Omohundro theory might yield ideas about new impact measurements that work with the AI's basic drives.

The AI safety research has attracted some very bright minds that are taking this issue seriously and working on creating a future with aligned AI. More specifically, in the development of impact measurements promising ideas have emerged. Although it is hard to say if this research is enough, and judging from what is at stake, why take the risk?

Bibliography

Amodei, D., Olah, C., Steinhardt, J., Christiano, P. F., Schulman, J., and Mané, D. (2016). Concrete problems in AI safety. *CoRR*, abs/1606.06565.

Armstrong, S. (2022). Why i'm co-founding aligned ai. [Online; accessed 7-June-2022].

Armstrong, S. and Levinstein, B. (2017). Low impact artificial intelligences. *CoRR*, abs/1705.10720.

Arulkumaran, K., Cully, A., and Togelius, J. (2019). Alphastar: An evolutionary computation perspective. *CoRR*, abs/1902.01724.

Bostrom, N. (2003). Ethical issues in advanced artificial intelligence.

Bostrom, N. (2012). The superintelligent will: Motivation and instrumental rationality in advanced artificial agents.

Bostrom, N. (2014). *Superintelligence: Paths, Dangers, Strategies*. Oxford University Press.

Carey, R. (2017). Incorrigibility in the CIRL framework. *CoRR*, abs/1709.06275.

Christiano, P., Leike, J., Brown, T. B., Martic, M., Legg, S., and Amodei, D. (2017). Deep reinforcement learning from human preferences.

Cotra, A. (2020). Forecasting tai with biological anchors.

Cotra, A. (2022). two year update on my personal ai timelines. [Online; accessed 20-October-2022].

Critch, A. and Krueger, D. (2020). AI research considerations for human existential safety (ARCHES). *CoRR*, abs/2006.04948.

Custard Smingleigh (2018). Tweet. [Online; accessed 01-06-2022].

Eysenbach, B., Gu, S., Ibarz, J., and Levine, S. (2017). Leave no trace: Learning to reset for safe and autonomous reinforcement learning. *CoRR*, abs/1711.06782.

FLI (n.d.). Existential risk. [Online; accessed 31-May-2022].

Galef, J. (2021). Humanity on the precipice (toby ord).

Good, I. J. (1965). Concerning the first ultraintelligent machine. *Advances in Computers*.

Grace, K., Salvatier, J., Dafoe, A., Zhang, B., and Evans, O. (2017). When will AI exceed human performance? evidence from AI experts. *CoRR*, abs/1705.08807.

Hadfield-Menell, D., Dragan, A. D., Abbeel, P., and Russell, S. (2016). The off-switch game. *CoRR*, abs/1611.08219.

Hadfield-Menell, D., Milli, S., Abbeel, P., Russell, S., and Dragan, A. D. (2017). Inverse reward design. *CoRR*, abs/1711.02827.

Hilton, B. (2022). Preventing an ai-related catastrophe. [Online; accessed 30-August-2022].

Hubinger, E., van Merwijk, C., Mikulik, V., Skalse, J., and Garrabrant, S. (2019). Risks from learned optimization in advanced machine learning systems. *CoRR*, abs/1906.01820.

Häggström, O. (2019). Challenges to the omohundro-bostrom framework for ai motivations. *Foresight*.

Häggström, O. (2021a). Ai, orthogonality and the müller-cannon instrumental vs general intelligence distinction. *CoRR*, abs/2109.07911.

Häggström, O. (2021b). *Tänkande Maskiner - Den artificiella intelligensens genombrott*. Fri Tanke.

Krakovna, V., Orseau, L., Martic, M., and Legg, S. (2018). Measuring and avoiding side effects using relative reachability. *CoRR*, abs/1806.01186.

Krakovna, V., Orseau, L., Ngo, R., Martic, M., and Legg, S. (2020). Avoiding side effects by considering future tasks. *CoRR*, abs/2010.07877.

Levinson, J., Askeland, J., Becker, J., Dolson, J., Held, D., Kammel, S., Kolter, J., Langer, D., Pink, O., Pratt, V., Sokolsky, M., Stanek, G., Stavens, D., Teichman, A., Werling, M., and Thrun, S. (2011). Towards fully autonomous driving: Systems and algorithms.

Mandhane, A., Zhernov, A., Rauh, M., Gu, C., Wang, M., Xue, F., Shang, W., Pang, D., Claus, R., Chiang, C.-H., Chen, C., Han, J., Chen, A., Mankowitz, D. J., Broshear, J., Schrittwieser, J., Hubert, T., Vinyals, O., and Mann, T. (2022). Muzero with self-competition for rate control in vp9 video compression.

McCarthy, J. and Hayes, P. J. (1969). Some philosophical problems from the standpoint of artificial intelligence. In Meltzer, B. and Michie, D., editors, *Machine Intelligence 4*, pages 463–502. Edinburgh University Press. reprinted in McC90.

McCarthy, J., Minsky, M., Rochester, N., and Shannon, C. (1955). A proposal for the dartmouth summer research project on artificial intelligence. *AI Magazine*, 27(4).

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. A. (2013). Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602.

Müller, V. and Cannon, M. (2021). Existential risk from ai and orthogonality: Can we have it both ways? *Ratio*.

Omohundro, S. (2008). The nature of self-improving artificial intelligence. *Proceedings of the First AGI Conferance*.

OpenAI (2018). Key concepts in rl. [Online; accessed 30-May-2022].

Ord, T. (2020). *The precipice : existential risk and the future of humanity*. New York Hachette Books.

Orseau, L. and Armstrong, S. (2016). Safely interruptible agents. *MIRI*.

Russell, S. (2019). *Human Compatible: Artificial Intelligence and the Problem of Control*. Viking.

Russell, S. and Norvig, P. (2021). *Artificial Intelligence - A Modern Approach, Fourth Edition*. Pearson.

Saisubramanian, S., Zilberstein, S., and Kamar, E. (2020). Avoiding negative side effects due to incomplete knowledge of AI systems. *CoRR*, abs/2008.12146.

Shah, R. (2020). Forecasting transformative ai timelines using biological anchors. [Online; accessed 12-June-2022].

Silver, D. and Hassabis, D. (2017). Alphago zero: Starting from scratch. [Online; accessed 30-May-2022].

Silver, D., Huang, A., Maddison, C., Guez, A., Sifre, L., Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. (2016). Mastering the game of go with deep neural networks and tree search. *Nature*.

Silver, D., Singh, S., Precup, D., and Sutton, R. S. (2021). Reward is enough. *Artificial Intelligence*, 299:103535.

Soares, N., Fallenstein, B., Yudkowsky, E., and Armstrong, S. (2016). Corrigibility. *MIRI*.

Turing, A. M. (1951). Intelligent machinery, a heretical theory. *Philosophia Mathematica*.

Turner, A. M. (2019). Optimal farsighted agents tend to seek power. *CoRR*, abs/1912.01683.

Turner, A. M., Hadfield-Menell, D., and Tadepalli, P. (2019). Conservative agency via attainable utility preservation. *CoRR*, abs/1902.09725.

Turner, A. M., Ratzlaff, N., and Tadepalli, P. (2020). Avoiding side effects in complex environments. *CoRR*, abs/2006.06547.

Wikipedia (2022). Artificial intelligence — Wikipedia, the free encyclopedia. [Online; accessed 30-May-2022].

Wikipedia (2022a). ENIAC — Wikipedia, the free encyclopedia. [Online; accessed 19-September-2022].

Wikipedia (2022b). Goodhart's law — Wikipedia, the free encyclopedia. [Online; accessed 01-June-2022].

Yudkowsky, E. (2013). Intelligence explosion microeconomics. *MIRI*.

Yudkowsky, E. (2016). Ai alignment: Why it's hard, and where to start.