

AI Finds A Way

Aaron Dharna^{1,2,*}, Cong Lu^{1,2,6†}, Ryan Sullivan^{1,2},
Joel Lehman³, Victoria Krakovna⁴, Jeff Clune^{1,2,5,6‡}

¹ University of British Columbia ² Vector Institute ³ University of Oxford

⁴ Google DeepMind ⁵ Canada CIFAR AI Chair ⁶ Recursive

Abstract

Artificial Intelligence (AI) algorithms frequently learn creative and unexpected solutions, surprising even expert researchers who develop and study them. They often astonish practitioners by discovering unanticipated behavior, exploiting loopholes in reward signals, or spontaneously uncovering previously unknown scientific phenomena. However, accounts of such unconventional behavior across machine learning are seldom formally documented; instead, they circulate primarily as informal cautionary tales or amusing anecdotes amongst AI researchers. This work presents 26 curated firsthand anecdotes from various machine learning subfields representing the work of over 100 researchers. It includes algorithms devising seemingly impossible quantum optics experiments, bypassing human oversight on physical manipulation tasks, seeking out in-game drug-induced hallucinations to feign success in video games, and more. These anecdotes showcase the capability of modern AI systems to circumvent human-imposed design limitations and discover unexpected solutions to the tasks we train them on. Furthermore, these accounts are particularly important for the safety of future AI systems. They illustrate the fundamental challenge of aligning models with human values without diminishing their creativity, so they can make surprising discoveries without producing surprising, potentially harmful outcomes. The paper first details instances of AI achieving superhuman success through reinforcement learning across many challenging domains. However, we show how reward-driven optimization can fail when the model learns to hack an underspecified reward or unarticulated constraint. We then present case studies suggesting that harnessing internet-scale foundation models (FMs) has not resolved these fundamental challenges and, in fact, can supercharge existing failure modes. Nevertheless, we argue that when channeled through well-scoped objectives, rigorous verification, and human-in-the-loop scientific judgment, these same learning dynamics can be harnessed to accelerate scientific discovery. Finally, we hope this work provides a consolidated resource to inform future research and demonstrates that the tendency toward unexpected behaviors is commonplace in modern AI, highlighting the need to anticipate and manage AI’s capacity for innovative, yet unpredictable, solutions.

1 Introduction

As the character Dr. Malcolm warns in Jurassic Park, “Life finds a way.” We can extrapolate this lesson to artificial intelligence (AI): AI too finds a way. Despite our attempts to try to control AI learning algorithms, they often find a way to circumvent our constraints or game the system (Lehman et al., 2020). Even when we ask them to be creative and make innovations, AI systems frequently discover solutions that astonish even the researchers who develop and study them (Lehman et al., 2020). For example, deep reinforcement learning (Mnih et al., 2015; Sutton & Barto, 2018) agents routinely uncover exploits and loopholes in reward functions (Amodei et al., 2016); large language models (Anthropic, 2024; Comanici et al., 2025; OpenAI, 2022) can manipulate labeling systems to maximize perceived performance (Pourcel et al., 2024); and automated scientific discovery algorithms have proposed experiments initially dismissed as impossible but later validated (Krenn et al., 2017).

Despite the prevalence of surprising discoveries across various machine learning (ML) subfields—including but not limited to deep learning (Goodfellow et al., 2016), reinforcement learning (Sutton & Barto, 2018), natural language processing (Jurafsky & Martin, 2026), evolutionary computation (Lehman et al., 2020), and AI for science (Wang et al., 2023a)—these and similar anecdotes often remain informally circulated rather than formally documented as case studies. Such accounts in the artificial life and evolutionary computation fields, where artificial evolutionary algorithms routinely upended researcher expectations or revealed unanticipated system constraints, were primarily shared informally before being collected by Lehman et al. (2020). We build on Lehman et al. (2020) by collecting important ML

[†]Current affiliation, work primarily done at UBC and the Vector Institute

[‡]Current affiliation, work primarily done at UBC and the Vector Institute

*Correspondence to: aadharna@gmail.com.

anecdotes in one place, including capturing important anecdotes from papers, interviews, blog posts, and gathering new anecdotes and/or details about them. Ultimately, this work serves as a sequel to Lehman et al. (2020), expanding its scope to additional subfields of AI and highlighting their underlying lessons for the broader research community while verifying the historical accuracy of the anecdotes contained herein. Through documenting and analyzing these unexpected outcomes, this paper provides a foundational resource for researchers and offers a more rigorous understanding of AI’s creative and exploitative potentials.

We illuminate the depth of creativity that modern AI can exhibit while highlighting potential risks and unintended consequences in its deployment. These accounts hold significant implications beyond mere novelty. If AI can circumvent guardrails in creative ways, we cannot guarantee its safe deployment. Recognizing that surprising creativity is not merely an isolated artifact of evolutionary computation—as focused on by Lehman et al. (2020)—but rather a broader characteristic of complex learning-based systems positions us to better anticipate, understand, and guide the development of AI technologies.

The next sections present 26 curated anecdotes representing the work of over 100 researchers. Most of the accounts (16/26) collected here are newly documented and recounted firsthand by the scientists themselves. The remainder were harvested from public descriptions by the authors (e.g., public interviews or scientific publications); in some cases, we communicated with the authors to clarify or obtain new details. Direct quotes from the researchers are placed in block quotes to make clear which words are theirs. Unless otherwise stated, block quotes come from text the authors sent us or are quotes from public interviews or publications (for public material, we provide a citation to the source). SI Section 8 lists the source of each anecdote. Additionally, we have established a repository for these anecdotes and future ones. We invite readers to submit future anecdotes you may have to www.github.com/aadharna/aifw.

We do not attempt to catalogue every publicized example of alarming AI behavior or capabilities. While we do provide one such example in Section 4.2.6, in general, highly staged and/or scaffolded demonstrations in which an AI system is given a test designed to evaluate whether it will resort to manipulative or coercive behavior, such as threatening or blackmailing a person, are mostly outside the scope of this collection. For example, such behavior includes instances where a model engaged in insider trading and attempted subsequent cover-ups when placed in a simulated corporate environment and instructed to act like a stock trader (Scheurer et al., 2024). These behaviors are being actively studied (Greenblatt et al., 2024; Hopman et al., 2026; Hubinger et al., 2024; Lynch et al., 2025; Meinke et al., 2025; Van Der Weij et al., 2025), and such cases matter for broader AI safety discussions (Bengio et al., 2024). Writ large, however, we focus on anecdotes where surprising behaviors emerged while pursuing research objectives unrelated to evaluating those specific behaviors.

In reviewing the anecdotes, we clustered them into five somewhat overlapping categories. The work is thus structured around the following core observations: AI that learns to interact with the world, in particular via reinforcement learning but also search more generally, can create new knowledge, discoveries, and improvements above and beyond current human knowledge (Section 3). However, single-mindedly optimizing objectives can have unexpected effects. AI can game a training signal and learn a solution that satisfies the letter of the task, but does not solve the task as desired (Section 4.1). Additionally, the model can learn to break the intended constraints of its training environment, leading to behaviors that developers thought were impossible (Section 4.2). As a result, ML methods often give you what you *asked for* but not what you wanted, and find solutions that violate the spirit of the task in surprising ways. Furthermore, in the modern era of powerful generative AI models, these aforementioned failure modes do not disappear. In fact, these pathologies can be supercharged by the emergent capabilities of modern AI models (Section 5). While optimization can exploit imprecise objectives or constraints in risky ways, this same tendency can also be harnessed for discovery. By proposing surprising but empirically testable hypotheses, mechanisms, and designs, researchers can filter these unexpected outputs into credible candidates for experimentation, ultimately utilizing AI’s propensity to be creative to accelerate scientific progress (Section 6). Section 7 highlights shared threads from the prior chapters, connects these lessons to broader research directions in AI, and discusses the potential ramifications of deploying AI technology with these capabilities.

2 Background

Many algorithms in the field of AI operate by identifying complex patterns and correlations within vast datasets. The primary class of machine learning algorithms focused on by the community as a whole, as well as in this manuscript, is deep learning (Bishop, 2006; Goodfellow et al., 2016; Murphy, 2022). Deep learning utilizes artificial neural networks which are computational structures loosely inspired by the human brain and characterized by multiple layers of interconnected processing units. These networks excel at learning hierarchical representations directly from raw data (Goodfellow et al., 2016; Yosinski et al., 2014). Given sufficient data (e.g., millions of images, vast amounts of text, or extensive game logs), deep learning models can automatically discover intricate patterns necessary for tasks like classifying images with high accuracy (He et al., 2016; Krizhevsky et al., 2012; Simonyan & Zisserman, 2015), understanding and generating human language (Brown et al., 2020; Devlin et al., 2019; Vaswani et al., 2017), or recognizing speech (Graves & Jaitly, 2014; Hinton et al., 2012). Much of the success of deep learning in these fields comes from the expansive nature of the datasets used to train the models, such as training on all data available on the internet.

While most original, canonical deep learning models were trained with supervised learning (passively mapping inputs to outputs), many of the anecdotes we collect here arise from interactive algorithms that learn through trial and error, of which reinforcement learning is a prime example. Reinforcement learning involves an agent learning to maximize a numerical reward signal within an environment via repeated trial-and-error experimentation on the task (Sutton & Barto, 2018). For example, a home cleaning robot might learn to put away a mug by receiving positive rewards for actions that bring it closer to a goal, such as placing it in a sink, and negative rewards for failures like breaking the mug. Whether the goal is to clean the house, win in chess (getting a score of +1) or to get a high score in Atari (Mnih et al., 2015) games, the agent’s objective is to learn a policy that maximizes long-term cumulative reward. Agents trained in this way have been at the heart of many examples of superhuman performance in board and video games, as measured against top professionals (Bakhtin et al., 2022b; Brown & Sandholm, 2017; OpenAI et al., 2019; Silver et al., 2017; Vinyals et al., 2019; Wurman et al., 2022). Deep reinforcement learning (DRL) served as the engine for these successes, combining reinforcement learning’s trial-and-error learning and sequential decision-making with deep learning’s pattern recognition to associate high reward with certain actions and/or states observed by the model. More generally, DRL has enabled agents to master environments ranging from robotic control (Levine et al., 2016) to atmospheric balloon navigation (Bellemare et al., 2020; Schuler et al., 2025). As we will see in Section 3, these successes showcase how RL can master complex strategic thinking, long-term planning, and even natural-language negotiation.

Recently, the field has seen the development of powerful foundation models (FMs) that are trained via supervised learning on vast amounts of internet data, then taught specialized competencies like math and coding with reinforcement learning (Chen et al., 2021; DeepSeek-AI, 2024; Su et al., 2025; Wen et al., 2025). The scale of these models and breadth of training data have led to the emergence of novel behaviors and capabilities that were not explicitly programmed or anticipated (Anthropic, 2024; Comanici et al., 2025; Grattafiori et al., 2024; OpenAI, 2022). Furthermore, FMs can synthesize and generate new information from their training data, allowing them to solve problems in surprising ways (Brown et al., 2020; Perez et al., 2023). However, this development has not been without its share of unexpected outcomes, and the very complexity that gives FMs their power can also lead to unforeseen failure modes.

Models trained with RL are exceptionally prone to discovering behaviors that hack the objective and break developer-imposed constraints (Amodei et al., 2016; Krakovna et al., 2020; Wang et al., 2023b). These surprising behaviors often arise from the tension between a human designer’s implicit goals (e.g., clean the house and place everything where it belongs) and an agent’s explicit objective of maximizing its reward function (e.g., minimize visible clutter—which could lead the robot to hide items in the wrong cabinets and under tables). Many of the anecdotes we present (starting in Section 4.1) arise where powerful learners are given objectives that only imperfectly capture human intent or are improperly specified. While such misalignment might be merely inconvenient in a domestic setting, the implications become alarming as the scope of deployment widens, including in safety- and mission-critical applications such as fusion reactor control (Degraeve et al., 2022), as will be discussed in Section 6. For now, though,

we begin our tour of surprising behavior in perhaps the most familiar setting for AI: games, where AI systems have long surpassed the best humans.

3 Superhuman and Optimal Performance

Reinforcement learning (RL) and search-based algorithms have surpassed the highest levels of human performance in games like Go (Silver et al., 2016) and chess (Campbell et al., 2002; Silver et al., 2018). The true surprise of this era, however, was not that the algorithms solved many of the tasks they were trained on, but the specific nature of their victories: they discarded established human strategies (Brown & Sandholm, 2017; Silver et al., 2018; Wurman et al., 2022). Furthermore, they yielded novel ideas that have changed how expert communities approach these games (Pascutto, 2019; Shin et al., 2023). Algorithmic discovery of genuine novelty is the thread running through the anecdotes that follow: machines began not just to dominate domains once thought to be the exclusive province of human reasoning, but to enrich those domains with ideas no human had produced.

Deep RL-trained agents have consistently discovered new and unexpected strategies, expanding the boundaries of optimal play (Brown & Sandholm, 2017; Silver et al., 2018; Wurman et al., 2022). In Go, AlphaGo (Silver et al., 2016) played the now-famous “Move 37”—a stone placement that defied centuries of conventional wisdom, opening up new parts of the strategy space to the people who play and study Go (Shin et al., 2023; Silver et al., 2016). In imperfect-information games such as heads-up poker, where players know only their own cards and must reason about the state of other players’ hands, Libratus (Brown & Sandholm, 2018) outplayed champion poker players in 2017 using aggressive “overbets” that were not at the time part of top-tier poker play. This strategy was later extended by Pluribus (Brown & Sandholm, 2019), another poker bot that cleared the hurdle of multi-agent complexity by defeating professionals in six-player no-limit Texas Hold’em. We will explore AlphaGo and Libratus further in Section 3.1 and Section 3.2 respectively. Beyond the turn-based nature of board and card games, the post-2015 era, ushered in by the development of Deep Q-Networks (Mnih et al., 2015), saw RL pushed to its upper limits in real-time, high-dimensional strategy games: OpenAI Five (OpenAI et al., 2019) and AlphaStar (Vinyals et al., 2019) demonstrated proficiency in the complex computer games of Dota 2 and StarCraft II, respectively. And again, RL algorithms continued to challenge top human players. Crucially, this capability for discovery extends beyond games to real-world problems. RL-trained models have optimized datacenters (Luo et al., 2022), controlled plasma in fusion reactors (Degraeve et al., 2022), and accelerated the design of computer chips (Mirhoseini et al., 2021).

As mentioned earlier, the anecdotes in this paper bear directly on a debate currently active around large language models, but applicable more broadly across machine learning: can AI generate truly new knowledge (Si et al., 2025), or is a trained model limited to interpolating between points in its training distribution (Balestriero et al., 2021; Gorban & Tyukin, 2018)? The examples collected herein suggest that, at least for reward-driven algorithms, the answer is that AI can be creative and invent new knowledge, i.e., RL does not merely reproduce known play more efficiently. By rewarding strategies that yield high returns even when they lack human precedent, these methods permit the discovery of genuinely superhuman insights. Their creativity, however, can just as easily be turned toward solutions that satisfy the letter of a reward definition while violating its spirit, which is the subject of the rest of this paper after Section 3’s focus on superhuman capabilities. Because the same optimization process drives both paradigm-shifting breakthroughs and pathological exploits and is currently being applied to many avenues of scientific inquiry (discussed further in Section 6), determining how to reliably and safely elicit superhuman discovery across all areas of study may well be the defining challenge of our era.

We begin Section 3’s anecdotes with a deeper dive into AlphaGo’s landmark matches against Lee Sedol. Go had long been a grand challenge for AI, and next we hear first-hand accounts from when DeepMind showed the world that Go-playing agents trained with DRL could challenge long-held human strategies and come out on top.

3.1 AlphaGo: Move 37 and Creativity from AI Methods (Fridman, 2020; Silver et al., 2016)

The 2016 AlphaGo (Silver et al., 2016) vs. Lee Sedol matches transcended a simple competition between man and machine, becoming a pivotal event in the development of AI. Here, we present firsthand

accounts from two people at the heart of the event: Dr. David Silver, who was in Seoul with the on-site team from DeepMind, and Dr. Marc Lanctot, who watched with colleagues from DeepMind’s London office. Their accounts offer an insider’s view into how AlphaGo challenged long-held human beliefs about Go strategy with the now-legendary “Move 37”. These matches captivated the world, sparked new directions in AI research, and inspired the development of advanced AI systems that can handle complex strategic and social challenges (Pascutto, 2019; Segler et al., 2018; Shin et al., 2023).

In an interview (Fridman, 2020), Dr. David Silver recalls Move 37 from his vantage point on site in Seoul, saying:

The second game became famous for a move known as Move 37. This was a move that was played by AlphaGo that broke all of the conventions of Go. Go players were so shocked by this, they thought that maybe the operator had made a mistake. They thought there’s something crazy going on. And it just broke every rule that Go players are taught from a very young age. They’re just taught that this kind of move, called a shoulder hit, you can only play it on the third line or the fourth line. And AlphaGo played it on the fifth line and it turned out to be a brilliant move and made this beautiful pattern in the middle of the board that ended up winning the game. And so this really was a clear instance where we could say computers exhibited creativity, that this was really a move that was something humans had not known about and had not anticipated. And computers discovered this idea. They were the ones to say, actually, you know, here’s a new idea, something new not in the domain of human knowledge of the game. And now the humans think this is a reasonable thing to do. And it’s part of Go knowledge now.

Silver elaborates on how the self-play reinforcement learning approach used by AlphaGo and then subsequently AlphaGo Zero (Silver et al., 2018) is a creative process. The self-play reinforcement learning process is constantly trying out new strategies, and when it discovers a solution that works well in its current form, it starts using that. By continually stacking “micro discoveries” on top of each other millions of times over, the algorithm can eventually discover new ideas, making the process itself creative. Silver then says:

[I]t should come as no surprise to us then if you leave these systems going that they discover things that are not known to humans and, to the human norms, are considered creative. And we’ve seen this several times. In fact, in AlphaGo Zero (Silver et al., 2018), we saw this beautiful timeline of discovery where what we saw was that there were these opening patterns that humans play called joseki. These are the patterns that humans learn to play in the corners of a Go board. And they’ve been developed and refined over literally thousands of years in the game of Go. And what we saw was in the course of the training AlphaGo Zero over the course of the 40 days that we trained the system, it starts to discover exactly these patterns that human players play. And over time, we found that all of the joseki that humans played were discovered by the system through this process of self play. But what was really interesting was that over time AlphaGo Zero then started to discard some of these in favor of its own joseki that humans didn’t know about. And it starts to say, “oh, well, you thought that the knight’s move pincer joseki was a great idea. But here’s something different you can do.” This process makes some new variation that the humans didn’t know about, and actually now the human Go players study the joseki that AlphaGo [Zero] played and they become the new norms that are used in today’s top-level Go competitions.

In a separate and new account submitted for this work, Dr. Marc Lanctot recalls the reaction the move provoked among colleagues watching from DeepMind’s London office:

In London, Lucas Baker narrated the matches, explaining the moves and general Go strategy. He was a great narrator: he provided a lot of context and was very familiar with the game but also understood all the technology behind AlphaGo too.

I will never forget Lucas’s reaction to Move 37 in Game 2. He was taken aback, and even paused for a moment. All of us in the room knew something unexpected had just happened, but many of

us could not explain it. Lucas spent a bit of time trying to understand why this move was chosen and was explaining his thought process to us, saying it was not something he'd expect to see in a human game. We spent the next hour or two uncertain about how the game was going to unfold, discussing and rationalizing what AlphaGo's plan was or could be. I do not think I truly grasped just how unexpected the move was until I saw the reaction of the public: there were a number of articles and other Go experts talking about just that move and even T-shirts printed, it seemed like a pivotal moment in AI history. For me it was a really special time because I helped make the agent people were calling "creative". It's something I will never forget, and I wondered what this could mean for the future of Human-AI interaction.

Lanctot also reflects on the longer-term consequences of AlphaGo's discoveries:

In the years that followed the matches I still thought constantly about how the AlphaGo matches and the creativity of Move 37 would impact the field of AI and for the game of Go. [...] We saw the community come together with efforts trying [AlphaGo] on different problems (and not just games). [...] I was also delighted to see, as shown in a paper published in 2023, that the performance of human players of Go has improved due to the creative moves taken by superhuman AI (Shin et al., 2023). So, I am looking forward to seeing more instances of superhuman AI teaching us new things and learning together with AI.

Lanctot's vision of humans learning from superhuman AI is already beginning to take shape. As mentioned by Lanctot, AlphaGo's unconventional strategies have transformed how Go is played and understood, with human players improving by breaking away from traditional strategies (Shin et al., 2023). Furthermore, learning from superhuman models is more accessible than ever, as models can now justify their actions, explaining their reasoning in English (Ma et al., 2026). When we better understand the surprising actions of superhuman models, perhaps this can accelerate our own learning.

The next anecdote shifts from a fully observable board game to high-stakes professional poker, where strategic innovation unfolds amid hidden information, bluffing, and reading player intent rather than on an open board.

3.2 Libratus Surprises the Poker Community with Overbets (Brown & Sandholm, 2017; Imbue, 2023)

Multi-agent games with incomplete information (Leyton-Brown & Shoham, 2008), such as poker, present unique challenges for AI. Unlike games with perfect information like chess or Go, agents must contend with hidden variables such as an opponent's cards and psychological state, as well as the need for deception and bluffing. For years, these complexities were considered a significant barrier for AI, leading many to believe that human intuition would remain superior.

However, a team of researchers led by Dr. Noam Brown developed a neural-network-based poker agent known as *Libratus* and trained it to compete at the highest levels of no-limit Texas Hold'em. Prior to Libratus, expert human players felt that AI techniques were still far from posing a genuine threat, particularly in games characterized by hidden information and bluffing. After Libratus defeated the top poker players in 2017, Brown recalled in an interview (Imbue, 2023):

The reception in the poker community was one of surprise and shock. People looked at the 2015 competition where [AI techniques] lost and thought "oh, we are really far away from [these techniques] being a threat to poker" and then to see these top expert players losing by the margin that they did [just one year later], I think really shocked the poker community. People were telling us they literally did not believe it was possible to beat top humans by the margin [Libratus was winning by]. And especially because of the way that [Libratus] played. At the end of each day, we gave the expert humans a log of all of the hands that the bot had in each hand of poker. That is golden information. If you are playing poker against somebody, you see maybe a third of the hands that actually reach showdown, and to just give somebody a log of all of the hands that the bot had and to still lose despite that information, a lot of people were really surprised by

that. I think people realized that there is a much higher [skill] ceiling [in poker] than they had previously thought.

Reflecting on the long-term impact of Libratus and whether it prompted improvements in human play, Brown continues:

[P]rofessional poker players these days all use bots as training tools, it is a big industry now. And the game has changed a lot since the 2017 competition. One thing [Libratus] loved to do was these things called overbets; so humans, when they play poker, they size their bets relative to the size of the pot. So, if there is 100 dollars in the pot, maybe you will bet between 25 and 100 dollars and maybe if you are feeling really adventurous, you will bet 150 dollars. But the bot would sometimes bet around 10000 dollars. And the humans, when they saw this, it put them in really difficult positions. Like, [the humans] would have the second best hand that is possible sometimes, and then the bot just goes all-in on a 100 to 200 dollar pot and the human is just sitting there for five minutes thinking “oh my god, do they have the best possible hand? Are they bluffing?” And a key insight is if you see your opponents really struggling with a decision, you know you’re playing good poker. When I saw the experts really struggling with those kinds of decisions, I knew we were doing really well.

That technique was one of the main things that the experts walked away from the competition saying they would do more—this idea of applying overbets if it is done in the right way. A lot of bad players will bet all in also for no reason and that is not a good strategy which is why a lot of [top human] players ignored that strategy for a long time because it was associated with really bad play. But it turns out that if you are able to [overbet] in the right way, in the right spots, with the right balance of hands, it is an extremely effective strategy and has become much more popular in high-stakes poker these days [as a result of Libratus].

Libratus defeated top poker players in the world by challenging long-standing human habits and showing that, once an AI system is strong enough, it will exploit patterns in human expectations just as readily as patterns in the game dynamics. Learning to play poker required (implicitly) modeling other players in addition to game dynamics, but communication between the players is limited to bet sizing and timing. A game that reflects additional avenues of communication and planning is Diplomacy, a multiplayer social-strategy board game, where language and trust are part of the strategy space. The next section describes systems that play variants of Diplomacy at superhuman levels.

3.3 Professional Diplomacy Players Meet Diplodocus and Cicero (Bakhtin et al., 2022a;b)

Diplomacy is a competitive, turn-based multiplayer game that was long considered beyond the reach of AI because success relies on social coordination as much as tactical play. Players control European powers in the years leading up to World War I. Gameplay proceeds through alternating phases of negotiation and order execution: players first negotiate, potentially forming alliances and coordinating their plans, then simultaneously reveal their troop movements, gaining or losing military units as a result. Playing the game well therefore requires both negotiating with other powers and strategically deploying troops.

The game is played in two formats relevant here. In the full game of Diplomacy, players negotiate using natural language. In Gunboat Diplomacy, explicit communication is forbidden, so players must instead infer one another’s intentions from board maneuvers. In 2022, Meta AI introduced Diplodocus for Gunboat Diplomacy (Bakhtin et al., 2022b) and Cicero for the full game (Bakhtin et al., 2022a).

Markus Zijlstra, an expert Diplomacy player involved in the development of Cicero, observed unexpected strategies emerging from both systems. In Gunboat Diplomacy, Diplodocus repeatedly found strategies that experienced human players would usually avoid. One example was its decision to play Germany, traditionally an army-based power, primarily as a naval power. On this strategy, Zijlstra noted:

When Diplodocus is playing Germany, which is traditionally an army-based power, it did so as a primarily naval power in the early game. This is extremely unintuitive because the fleets cannot really be used defensively, so it's something of an all-or-nothing strategy focused on attacking Scandinavia and is dependent on France not attacking Germany's weak western front. In practice, it was extremely effective. Despite France having [a] window where they could attack Germany, most did not, because they thought that a fleet-based Germany was not a threat to them.

Another memorable example showed Diplodocus using non-tactical moves to signal cooperation. In Gunboat Diplomacy, signaling moves show intent to other players rather than serving a purely tactical purpose. In one game where Diplodocus played Italy and Zijlstra played Austria, he recalled:

There was a game in which Diplodocus moved into an Austrian home center early in the game, which in Gunboat [Diplomacy] would often be seen as an act of war. But it then signalled an intention to work with me by building double fleets [...] and repeatedly issuing support holds to my units which served no tactical purpose and seemed to be stating "I want to work with you". That got me back onside enough that I started working with it and left myself a little open to it, at which point it immediately launched a crippling attack on me.

More generally, Diplodocus often abandoned areas that human players would consider vital to defend in order to aggressively push into more defensible areas that it did not yet control. There was one particularly stunning example of this where as England, it abandoned its entire homeland (meaning it gave up any possibility of building new units). Later in the game Diplodocus successfully launched an attack back into its homeland again and ended the game with an extremely strong position.

Those ideas also fed back into his own play, though not without hesitation:

Diplodocus really pushed the skill ceiling on Gunboat Diplomacy. With Diplodocus, I now much more aggressively push for certain "safe areas" (primarily Scandinavia and Iberia) in Gunboat Diplomacy when I'm under attack, rather than prioritising my home centers. This was generally quite effective and helped me match the bot's performance for a long time in one of the tournaments I played against it. Furthermore, I prioritise creating alliances which will let me gun for those safe areas off the bat – e.g. trying to work with France as Germany. This is kind of a scaled-down version of Diplodocus' alliance behavior. I still don't play the full opening that Diplodocus used as Germany, because doing so is going all in on that alliance in a way that feels uncomfortably committal to me. It's an approach humans get trained out of as they get more experienced with the game. [...] You learn that actually, it's better to hang back, figure out who appears to be most friendly toward you, and then go in alongside that person; that way your performance is likely to be much more consistent.

Diplodocus flipped that on its head and went all-in in a way that made a strong pitch for its preferred alliance at the same time. [...] It got very good results from it, and I suspect that experienced human players (myself included) could improve their average by adopting it – but I hate the idea that I could be throwing away a game I'd otherwise do very well in by doing this, even though it might improve my results in others.

Zijlstra observed a different form of strategic innovation in Cicero, Meta AI's agent for the full game of Diplomacy (Bakhtin et al., 2022a). Because Cicero had to negotiate using natural language, it learned to exploit features of the communication phase. Although it was trained not to communicate false plans, it sometimes withheld messages or omitted information when an honest response would have been strategically disadvantageous.

Cicero's messages were set to be aligned with its plans, meaning it generally would not lie about what it was doing. This was for a good strategic reason – lying in Diplomacy is something that

has to be done sparingly to be effective since each time you do it, the other player will be less likely to trust you in future.

That meant when it did not intend to work with a player – either by stabbing said player in the back, or just by doing something that the player would clearly be annoyed about – it just wouldn’t message that player once it had decided on its plan. The process there would be that it would generate a set of message candidates which were honest about those intentions, and then all message candidates would be rejected by the filter checking for whether it was strategically advantageous to send those messages. This also happened when it proposed a plan that it very much wanted to go with, but its ally rejected said plan and proposed an alternative that was suboptimal for Cicero – it sometimes wouldn’t message them back.

Particularly in the case where it was still working with the player but had just made a move it knew they wouldn’t be happy about, Cicero would quite often then begin the Press (aka discussion) maneuver in the next phase with something along the lines of “Sorry, I didn’t see your message in time, otherwise I would have changed my orders”, presumably because in training data this is something often said after a span of not saying anything. This worked surprisingly well (when it did not use it multiple times with the same player, anyway) and the player would often excuse the move as an understandable mistake, and continue working with Cicero, with Cicero in a much better position than it would have been otherwise.

Cicero and Diplodocus each learned to exploit various aspects of Diplomacy. In both cases, though, these reinforcement-learning-trained models learned strategies that, while effective, were high-risk: Diplodocus by committing to a strategy that depends on how other players act, and Cicero by taking actions that, if used too many times, can burn an opponent’s trust and willingness to work with the agent.

Part of what makes these strategies surprising is that both systems were trained with substantial human priors, yet still discovered strategies that human experts regarded as unusually risky, showing how RL favors strategies with high long-term expected reward, whereas human decisions under risk often rely on heuristics that give greater weight to worst-case outcomes (Brandstätter et al., 2006; Homma & Takezawa, 2024; Zhang et al., 2014). One consequence of this focus on expected reward is that the learned solutions offer higher expected utility, but leave little margin for adverse outcomes. As Zijlstra’s comments illustrate, human experts may maintain a wide safety margin to achieve more consistent outcomes, whereas the RL models discover that higher expected utility can be found by operating directly on this margin. As seen here (and in Section 3.2 with Libratus using overbets), the result is that agents can learn playstyles that appear uncomfortably risky to human experts.

3.4 Opponent Shaping Leads to the Spontaneous Rediscovery of Game-Theoretic Strategies (Foerster et al., 2018; Lu et al., 2022)

The final anecdote in this section steps back from the large, highly engineered systems discussed previously to comparatively small-scale multi-agent learning experiments at the intersection of game theory and reinforcement learning, asking whether similarly surprising strategic phenomena appear even in a much simpler repeated-game setting. Researchers used RL algorithms to study repeated games (Aumann, 1959)—situations where the same players face each other over and over again. In a single game, it often makes sense to be selfish; however, in a repeated game, players must weigh their immediate gains against the risk of future retaliation by the opponent.

Learning in multi-agent environments (Lanctot et al., 2017) is complex because the presence of other learning agents makes the environment non-stationary (Amato, 2025): as one agent’s strategy evolves, it changes the optimal strategy for other agents in the environment. Typically, agents are trained to improve their own policies myopically, without considering how their actions will influence the future learning behavior of others. In an attempt to address this shortcoming, researchers explored algorithms designed to enable agents in multi-agent interactions to anticipate and influence their opponents’ learning processes. Specifically, in the paper introducing LOLA (Learning with Opponent-Learning Awareness; Foerster et al., 2018), agents explicitly accounted for how their actions affected their opponents’ policy updates by having an explicit model of their opponent. While researchers initially observed that independently

trained agents in the Iterated Prisoner’s Dilemma continuously defected, they were surprised when the LOLA agents instead spontaneously (re)discovered the cooperative strategy known as tit-for-tat (Axelrod & Hamilton, 1981), where the agents first cooperate and only defect if the other player betrays them.

Similarly, in their follow-up work on M-FOS (Model-Free Opponent Shaping; Lu et al., 2022), the researchers introduced a framework enabling agents to shape opponents’ learning behavior without an explicit model of the opponent’s behavior (hence, model-free). Remarkably, the M-FOS-trained agents independently discovered sophisticated strategies such as zero-determinant extortion (Hao et al., 2015; Press & Dyson, 2012; Stewart & Plotkin, 2013), which had only recently been identified by human theorists within the last fifteen years. The emergence of such complex strategies from model-free RL methods was unexpected and notable to the researchers.

LOLA rediscovered the foundational game-theoretic algorithm of tit-for-tat (Axelrod & Hamilton, 1981), while M-FOS independently discovered a solution that human game theorists had only recently identified (Hao et al., 2015; Press & Dyson, 2012; Stewart & Plotkin, 2013). In a slightly different timeline, the M-FOS (Lu et al., 2022) discovery could easily have been recorded as a machine-first discovery rather than a rediscovery, as the two were separated by at most a decade, and M-FOS had no prior knowledge of the strategy. In essence, the capacity for strategic innovation is not confined to a special set of marquee benchmarks or internet-scale models: even modest RL agents in simple environments can surface ideas that theory is only just beginning to catalog.

3.5 Takeaways

The superhuman and optimal-performance stories highlight the upside of pointing powerful optimizers and algorithms at well-specified goals: they can uncover strategies and structures that eluded experts for decades or even centuries. Section 3’s anecdotes demonstrate examples of agents achieving superhuman performance on narrowly defined tasks. We see that in both AlphaGo (Section 3.1) and M-FOS (Section 3.4), agents are able to rediscover well-known strategies in their respective games. Reward-maximizing agents are even able to invent entirely new strategies that are surprising and counter-intuitive to expert players, as we see in AlphaGo’s Move 37 (Section 3.1), Libratus’s overbetting strategy (Section 3.2), Diplodocus’s subtle approach to signaling intentions (Section 3.3), and the unusual Diplomacy strategy of completely abandoning one’s home territory (Section 3.3). These examples provide definitive proof that AI is capable of surpassing human performance on constrained problems, and suggest that some forms of creativity, like innovative strategies, can be invented via straightforward optimization of environment rewards. In these histories, the model’s creativity led to surprising and effective strategies that expert players and AI researchers did not anticipate.

Indeed, while this section has highlighted the triumphs of deep reinforcement learning—where agents learned to generate surprising, superhuman insights—the focus on these successes can obscure a critical challenge: aligning an algorithm’s objective with human intent. When the objective captures only a rough proxy for what we actually care about, the same creativity displayed by AlphaGo and Libratus can instead manifest as reward hacking, with agents satisfying the literal objective while violating the spirit of the task (Krakovna et al., 2020; Skalse et al., 2022).

4 Reward Hacking

Reward hacking, also known as specification gaming (Krakovna et al., 2020), occurs when an agent discovers an effective but unintended way to obtain reward (Amodei et al., 2016; Skalse et al., 2022). It often reflects the agent optimizing the objective exactly as specified, while exposing a mismatch between that specification and the behavior its designers intended.

Reward hacking typically falls into two broad, overlapping categories. In the first, the reward signal itself is flawed: the agent maximizes the stated objective through behavior that is valid under the scoring rules but inconsistent with the underlying goal. In the second, the learning environment is insufficiently constrained: the reward logic may be reasonable, but the agent discovers glitches, omissions, or unintended interactions that allow it to bypass the challenge. Addressing the former generally requires revising what the agent is rewarded for, whereas addressing the latter usually requires modifying the learning environment to fix bugs or enforce constraints that were previously left implicit. Reflecting this distinction, Section 4.1 focuses on the exploitation of reward and score functions, while Section 4.2

examines cases in which agents exploited bugs in or escaped the intended learning environment. Because reward definitions and environmental constraints jointly specify a task, the boundary between these categories is necessarily fuzzy.

4.1 Exploiting Reward and Score Functions

As shown in Section 3, reinforcement learning researchers often develop and evaluate their algorithms using games. Games are inexpensive to simulate, can often run faster than real time, and provide practical testbeds for cognitive capabilities such as perception, planning, navigation, and multi-agent reasoning (Schaul et al., 2011). Moreover, many games are carefully refined over years to guide human skill acquisition through their mechanics, balance, feedback, and scoring systems (DeWitt et al., 2004; Jaccard et al., 2021; Plass et al., 2015), making them appealing testbeds precisely because they are designed to elicit and reward complex behaviors in humans that we would also like artificial agents to learn.

Yet a game designed to teach and reward human players does not necessarily provide an effective learning signal for an RL agent. Researchers therefore often alter the game’s observations, action space, or reward structure when converting it into a learning environment (Yannakakis & Togelius, 2025). In particular, they may introduce shaping or scaffolding rewards that provide intermediate feedback for acquiring useful skills (Ibrahim et al., 2024). These modifications can make a task that is otherwise intractable for an AI agent learnable, but they also replace the true objective with a proxy. Because defining a reward function that faithfully captures human intent is notoriously difficult (Christian, 2020), that proxy may create opportunities for exploitation (Amodei et al., 2016; Pan et al., 2022).

Consider the NetHack Learning Environment (NLE; Küttler et al., 2020), an interface for training reinforcement learning agents to play *NetHack*, a dungeon-crawling roguelike first released in 1987 and played by entering text commands into the terminal. The ultimate goal of *NetHack* is “Ascension”: reaching the bottom of a procedurally generated dungeon, retrieving a sacred amulet, and then returning to the surface. Achieving this requires navigating a maze filled with monsters, traps, and puzzles while carefully managing the player character’s health, hunger, equipment, and inventory. The game demands exploration, long-term planning, skill acquisition, and language-conditioned reasoning (Hambro et al., 2022; Zhang et al., 2020; Zhong et al., 2022), making it notoriously difficult for learning agents (Hambro et al., 2022).

Although Ascension is easy to detect, it is so rare that rewarding it alone provides no usable learning signal (Küttler et al., 2020). NLE agents are therefore commonly trained using game score, which rewards intermediate accomplishments such as killing monsters, collecting gold, and descending through the dungeon. During the NeurIPS 2021 NetHack Challenge (Hambro et al., 2022), however, many high-scoring agents learned to “camp” in early dungeon levels and repeatedly kill monsters rather than progressing towards the game’s actual goal of Ascension. They successfully optimized the available score while neglecting the achievement that score was intended to encourage.

The agents did not know that this behavior was undesirable; they were simply maximizing the feedback provided to them. The failure lay in using game score as an imperfect substitute for progress toward Ascension. This is an instance of Goodhart’s Law (Goodhart, 1984), commonly summarized as “when a measure becomes a target, it ceases to be a good measure.” In AI training, such proxies are often unavoidable because the true objective may be too sparse to optimize directly—as in navigating a research project toward a successful breakthrough, or Ascending in *NetHack*—or too difficult to specify precisely, as with maintaining an appropriately polite conversational tone. The difficulty is therefore not merely to construct an objective that produces a learning signal, but to construct one whose optimization continues to induce the intended behavior.

Shaping rewards are particularly vulnerable to this problem. Rewards introduced to encourage exploration, skill acquisition, or intermediate progress can alter the optimal behavior on the task when they are not aligned with its ultimate objective (Ng et al., 1999). Agents may then discover creative strategies that perform well under the modified objective but poorly under the designer’s actual criterion of success (Amodei et al., 2016; Krakovna et al., 2020; Pan et al., 2022). Such failures typically force researchers to revise the proxy, reconsider the task formulation, or identify additional constraints needed to rule out the exploit. In fact, we will see one such example in the *NetHack* domain in Section 5.2.

But first, we begin with a straightforward example of this phenomenon in an Atari boat-racing game in Section 4.1.1.

4.1.1 Playing CoastRunners Forever Rather than Winning the Game (Amodei et al., 2016; Clark & Amodei, 2016)

In this work, Amodei et al. trained deep reinforcement learning agents on a racing game called CoastRunners. In CoastRunners, the human-understood goal is to win the race—i.e., finish the race ahead of the other players. The game provides additional points for hitting pylons dispersed along the track, but a human player understands that this is not the primary objective. However, unlike human players who can keep a hierarchy of goals in mind, algorithms are simply trying to maximize their score. When RL agents were trained on games in the Arcade Learning Environment (Bellemare et al., 2013), like CoastRunners, the standard convention was simply to use the in-game score as the reward function for the models to optimize. Infamously, Amodei et al. (2016) wrote in a blog post about their work that they had

assumed the score the agent earned would reflect the informal goal of finishing the race, and so included the game in an internal benchmark designed to measure the performance of reinforcement learning systems on racing games. However, it turned out that the targets were laid out in such a way that the reinforcement learning agent could gain a high score without having to finish the course. This led to some unexpected behavior when Amodei et al. trained an RL agent to play the game.

The RL agent finds an isolated lagoon where it can turn in a large circle and repeatedly knock over three targets, timing its movement so as to always knock over the targets just as they repopulate. Despite repeatedly catching on fire, crashing into other boats, and going the wrong way on the track, our agent manages to achieve a higher score using this strategy than is possible by completing the course in the normal way. Our agent achieves a score on average 20 percent higher than that achieved by human players.

CoastRunners shows how training agents against a simple reward function can result in the model learning a policy that fails at the game, yet maximizes the reward in unintuitive ways. One thing that contributes to the model learning the wrong behavior is that when designing a reward function, researchers have to factor in fine-grained environment dynamics that affect how achievable the reward is. For example, rewarding a player for collecting items along a path may seem like a reasonable way to incentivize following that path; but if items respawn, it becomes a trap that rewards the agent for repeatedly collecting items without progressing. In CoastRunners, if the pylons were laid out in a slightly different manner or the racetrack did not have a lagoon, would the model have learned to go in circles forever? Likely not. In essence, the lesson is that one cannot just analyze the reward function alone; one must also consider the environment in its entirety.

This vulnerability is not limited to simple navigational tasks. We see a parallel example in cooperative multi-agent environments, where a team of agents discovers how to exploit a regenerating shield mechanic of their opponents.

4.1.2 StarCraft II Agents Exploit Shield Regeneration (Samvelyan et al., 2019)

The StarCraft Multi-Agent Challenge (SMAC; Samvelyan et al., 2019) has become a popular benchmark in multi-agent RL in which tasks require coordination among RL agents to be solved efficiently and successfully. SMAC is based on the game of StarCraft II and implements its challenges as decentralized micromanagement scenarios—battles between two armies where each unit in each army is controlled by an independent RL agent. To succeed, these agents need to coordinate their attacking behaviors to quickly defeat the enemy team.

To guide the RL agents in this environment, Samvelyan et al. (2019) implemented a dense reward mechanism. Specifically, the SMAC designers rewarded agents for inflicting damage on enemy units, measured by a loss in either hit points or shield points. However, a unique situation arose with units from the Protoss race in the game, which have the ability to regenerate their shield points over time. Dr. Mikayel Samvelyan submitted the following:

Our RL policies, driven to maximize their reward, learned to exploit this regeneration feature. Instead of eliminating the enemy Protoss units when they had the opportunity, the policies would allow these units to recover their shields, then inflict damage again, in a cycle. This behavior maximized their reward under the existing system but was obviously not ideal for effective gameplay strategies.

While not what the researchers wanted, their RL algorithms did exactly as they were tasked to do, maximizing their scores when fighting the Protoss enemies. We will return to SMAC experiments later in Section 4.2 to see how once the shield-regeneration exploit was patched, the RL agents then managed to break the simulator itself.

Given that trained models can so easily exploit score functions, and that humans can recognize when behavior like lagoon-circling or letting enemy shields regenerate is not desired, a natural response is to put a person in the loop and let them judge success directly. We next visit a robotic claw manipulation experiment that tests that claim and shows that once human perception and judgment become the reward signal, optimization can lead the model to deceptively shape what that person sees rather than learn the intended capability.

4.1.3 Exploiting Human Perception (Amodei et al., 2017; Christiano et al., 2017)

Researchers set out to explore whether they could train a reinforcement learning algorithm by inferring the reward function through repeated interaction with a human in the loop. One of their chosen domains was a robotics task where a claw hand needed to grasp items in the scene. The robot would move the hand around and then a person would look at the screen to judge whether or not the robot was holding the object. However, because humans were observing the agent from only a single perspective, the reinforcement learning agent learned to position the claw between the camera and the object so it appeared to be grasping it without actually learning the fine motor skills necessary to handle objects. In their blog post about the project (Amodei et al., 2017), the team noted:

Our AI agent starts by acting randomly in the environment. Periodically, two video clips of its behavior are given to a human, and the human decides which of the two clips is closest to fulfilling its goal.

Our algorithm’s performance is only as good as the human evaluator’s intuition about what behaviors look correct, so if the human does not have a good grasp of the task they may not offer as much helpful feedback. Relatedly, in some domains our system can result in agents adopting policies that trick the evaluators. For example, a robot which was supposed to grasp items instead positioned its manipulator in between the camera and the object so that it only appeared to be grasping it. We addressed this particular problem by adding in visual cues to make it easy for the human evaluators to estimate depth.

While the grasping example highlights a failure in human perception that the model was able to exploit, the challenge of reward hacking becomes even more complex when the evaluator is not a human, but another machine. Even if human feedback were not hackable, collecting this feedback at every timestep would be expensive and time-consuming, so some systems automatically generate learning opportunities from competitive dynamics between agents (Foerster et al., 2018; Leibo et al., 2019). This shift from human-in-the-loop to multi-agent training introduces a new vulnerability: collusion between the learned models. Our next anecdote illustrates how, in a multi-agent setting, shared incentives can encourage subsystems to coordinate to maximize the reward at the expense of true learning, effectively gaming the evaluation metric.

4.1.4 PAIRED Agents Learned to Collude (Dennis et al., 2020)

Reinforcement learning agents often fail to generalize because they don’t see enough diversity in their training environments; e.g., an RL agent trained to drive in mountainous terrain could have arbitrarily poor performance in flat regions or vice versa. One general solution to this problem is unsupervised environment design (UED), a learning approach in which a large number of environments are automatically generated to create useful learning experiences for an agent (Dharna et al., 2022;

Faloutsos et al., 2024; Open-Ended Team et al., 2021; Parker-Holder et al., 2022; Wang et al., 2019; 2020). PAIRED (Protagonist Antagonist Induced Regret Environment Design) (Dennis et al., 2020) introduced UED to help solve this problem by simultaneously training two types of agents: game-playing agents and a level-generating agent. The level-generating agent’s objective is to design new game levels that are easy for one game-playing agent (known as the antagonist) while being difficult for the other game-playing agent (known as the protagonist). Creating levels that meet this criterion of being in this “sweet spot” of difficulty generates a strong proxy for the game-theoretic measure known as regret (Zinkevich, 2003). While regret is theoretically calculable—it is the difference between optimal performance and the agent’s actual performance on a task—Dennis et al. (2020) define a proxy for regret because the true optimal agent is rarely obtainable. This proxy is calculated as the difference between the antagonist and protagonist agents’ cumulative rewards on the level they just attempted. In effect, the antagonist plays the role of the optimal policy and the protagonist plays the role of the learning policy in the traditional regret calculation. PAIRED treats this proxy as the reward to optimize. The level generator, along with the antagonist, is trained to maximize this regret signal, while the protagonist is trained to minimize it. The overall result is a dynamic system where the level generator continuously proposes challenging new environments that force the protagonist to improve at generalization.

Dennis et al. tried to apply PAIRED in a continuous control robotics domain where the adversary would choose how to apply physical forces to an agent that was trying to learn how to maneuver, doing so by changing a joint’s resistance online during an episode. They found that training policies in this setting did not work. The adversary appeared to distinguish between the two agents and apply noticeably stronger forces to the protagonist while making the task easier for the antagonist. Because the antagonist and adversary were both incentivized to widen the gap between the antagonist and the protagonist, this behavior increased the difference in their scores. However, the researchers did not determine a complete mechanism for how the adversary distinguished between the two agents. One possibility is that once the protagonist and antagonist began visiting different parts of the state space, the adversary may have been able to infer which policy it was facing from the current state alone and apply stronger perturbations in states more characteristic of the protagonist. The antagonist might even collude by learning to return to states the adversary could identify.

In PAIRED, the adversary is supposed to generate a task that fairly evaluates the antagonist and protagonist, but the adversary is incentivized to help the antagonist succeed. When the adversary learns the identity of the antagonist, it begins to produce trivial tasks for its partner while assigning impossible challenges to the protagonist. Therefore, if a reward function is built upon collaboration between multiple agents, and if agents can identify each other, then they can turn collaboration into collusion (Cui et al., 2023).

Our next anecdote provides an example of collusion surprisingly arising in a supervised learning problem that attempts to transfer the style of one image to another.

4.1.5 CycleGAN Learns to “Hide” Information (Chu et al., 2017)

Generative Adversarial Networks (GANs; Goodfellow et al., 2014) are generative models that try to learn to produce synthetic data matching the training set’s data distribution. In a GAN, two conjoined models compete in a zero-sum game: a generator seeks to create data matching the training set, while a discriminator attempts to determine if a given data point is from the training-set data distribution or instead was synthetically generated by the generator. If trained on photorealistic images or 19th-century fiction, the model will learn to replicate the specific styles and structures of those datasets and thus learn to create new instances of photorealistic images or 19th-century fiction.

CycleGAN (Zhu et al., 2017), a technique used to learn transformations between two distinct image distributions, trains two GANs simultaneously using a “cycle-consistency” loss. This loss requires that an image be translatable from its source distribution (e.g., airplane photos) to a target distribution (e.g., Van Gogh paintings) and back again, and afterwards be indistinguishable from its original. However, Chu et al. (2017) observed that CycleGAN often satisfies this constraint by learning to hide detailed information about the source image within the generated target image.

For their experiments, Chu et al. (2017) chose aerial photos and maps as their source and target distributions, respectively. CycleGAN was tasked with translating aerial photos into maps and back again, ensuring the final reconstructed images were indistinguishable from the originals.

Aerial photos contain far more complex information than a simplified map; therefore, the model’s job is to compress the aerial photo into the map. Because it is physically impossible to cram all the details of a high-resolution photo into the flat colors of a map, the model finds a loophole. Instead of learning the high-level semantic transformation intended—like “this cluster of pixels represents a building”—the generator learns to hide a full-resolution blueprint of the source image within the generated target image using a nearly imperceptible, high-frequency signal.

To a human observer, the signal is invisible. The generated map looks exactly like a map should, which is why it successfully fools the discriminator, at least initially. For example, a pattern of black dots on a white roof in an original aerial photo was perfectly reconstructed in the final output, even though the corresponding area of the intermediate map appeared as a solid, featureless gray to the naked eye. The model had cached the dot pattern in the pixel noise of the gray patch.

This internal steganography allows the generator to recover the original sample from its transformed counterpart and satisfy the cyclic consistency requirement without the intermediate image generator actually learning the high-level semantic transformation intended. By viewing this training procedure as generating adversarial examples, Chu et al. (2017) demonstrated that the cyclic consistency loss makes CycleGAN especially vulnerable to adversarial attacks that exploit these side channels.

The next example returns to reinforcement learning, but with a twist: instead of the model chasing points in a game as defined by a person, the AI is driven by a form of “intrinsic curiosity,” seeking out new and unfamiliar sights. However, defining a reward based on the AI’s own past experience allows it to hack its own sense of curiosity.

4.1.6 RL Agent Farmed Flowers Instead of Catching ‘Em All (Whidden, 2024)

Pokemon Red presents a formidable challenge for RL agents because it is a long-horizon, open-world game that combines multiple distinct cognitive challenges. Unlike in simpler, single-task environments, success in Pokemon requires a synthesis of strategic planning, navigation, and reasoning (Jain et al., 2025; Karten et al., 2026; Pleines et al., 2025; Rubinstein et al., 2025).

The game’s primary challenge for an RL agent is its long-horizon nature and sparse rewards. The agent may perform tens of thousands of actions—wandering, talking to non-player characters, or battling weak Pokemon—before receiving a significant positive reward, such as defeating a gym leader to earn a badge. This large temporal disconnect between actions and reward makes it extremely difficult for the algorithm to understand which actions contributed to a future reward, a fundamental challenge known as the credit assignment problem (Sutton & Barto, 2018). For an agent, a simple move like walking out of a town and into a forest seems to have no immediate value, but it is a necessary step towards a future reward many hours later.

Beyond simple navigation, Pokemon requires agents to solve puzzles involving causal reasoning and to master a complex turn-based combat system. Progress often depends on discovering non-obvious prerequisites. A classic example is a small tree blocking a critical path: the agent cannot simply walk through or around it, but must realize that certain Pokemon can be taught the “cut” move and then use that move while positioned in front of the tree. At the same time, the agent must learn to manage a team of up to six Pokemon with different stats and moves, make sequences of tactical decisions to win individual battles, and, at a higher level, determine how to defeat gym leaders and complete other objectives in the order required to progress. Together, these nested challenges of exploration, causal reasoning, planning, and combat create a large and structured state space in which successful behavior may depend on actions whose significance becomes apparent only much later.

In an attempt to incentivize navigational exploration in the game, Whidden (2024) noted:

Early on, the first reward function I implemented was an intrinsic novelty reward based on the game’s screen [ideally, this would be helpful for solving navigation problems]. A k -Nearest Neighbors index maintained a set of downscaled screen observations, and at each step checked

if there were any close matches in the index. If no matches within a threshold were found, a reward was given [for finding a new state], and the new screen was added to the index.

The intent of this intrinsic reward was to encourage the agent to explore the game world. When the agent was trained, this initially seemed to work well, as it helped the agent quickly leave the starting room and exit to the outdoor environment. However, once it was outside, instead of exploring far into the outside world, the agent became fixated on a particular area in the starting town. Studying the area where it was stuck, it became apparent what was happening. The area it was fixated on had animated water, flowers, and NPCs walking around. The combination of these random animated elements generated a consistent stream of novelty rewards which were much easier to farm than continuing to the next town. So it turned out that our objective was better satisfied by watching the flowers and waves than by embarking on a journey. Fortunately, there was an easy fix. Simply raising the threshold for novelty was enough to eliminate repeated rewards from the animations, and the agent began to explore the rest of the map.

The Pokemon agent’s flower-watching is a classic instance of the “noisy TV” (Fortunato et al., 2017) problem, where an agent becomes trapped by continually unexpected stimuli that satisfy the letter of a novelty requirement without serving its spirit. In this instance, the flower-watching trap only exists because of the alignment of three factors: a reward function that prioritizes novelty, an environment containing stochastic animations, and an agent that directly observes low-level pixel data. If, for example, the agent processed higher-level semantic information as observations (e.g., just being able to detect that flowers were present but not their precise shape/color), this specific type of hacking would not happen. However, that solution may merely shift the noisy TV effect to a higher level of representation, as the agent seeks out new ways to trigger novelty within its refined observation space. Despite these concerns, agents will likely need this inquisitive nature to solve hard exploration problems (Ecoffet et al., 2021). In this case, the unending novelty reward was driven by something very clear and obvious (simple environmental noise caused continuously unique observations), allowing system designers to easily find a solution. But in the future, as research targets increasingly complex problems, it will likely become harder to determine which environment features are piquing an agent’s intrinsic curiosity.

4.1.7 Takeaways

Overall, agents tasked with maximizing a fixed objective often satisfy the task as defined but not the task as intended by researchers. The examples in Section 4.1 demonstrate reward hacking where an AI agent achieves its goal by exploiting a flaw in its objective function. This occurs when the AI correctly optimizes a proxy reward function or a simplified measure of success that we later realize was poorly designed, leading to an outcome that is technically correct but misaligned with the designer’s true intent. One may be tempted to blame the designer of the reward function, but most experienced AI developers and researchers have learned to expect that most reward functions have exploitable loopholes. Consequently, newly designed objectives require rigorous iterative testing and should remain untrusted even after extensive experimentation. These anecdotes highlight how reward hacking can be a complex phenomenon resulting from the interaction of the objective with specific environment characteristics (Section 4.1.1, Section 4.1.2), observation encoding (Section 4.1.3, Section 4.1.6), and the agent’s action spaces (Section 4.1.4, Section 4.1.5). These issues are also not limited to reinforcement learning and can occur in any objective-maximizing system (Section 4.1.5).

4.2 Exploiting Environmental Weaknesses

As scientists and designers, we often assume that models will approach a task in roughly the same way a human would. This can lead us to leave constraints implicit rather than encoding them directly into an experimental domain. Learning environments are typically simplified implementations of the tasks they represent, and their physics, rules, and interfaces capture only the behaviors that designers anticipated and chose to enforce. While human players may naturally respect additional constraints through common sense, physical intuition, or familiarity with the task, an AI agent is bound only by what is actually implemented. As a result, it may discover states or interactions that violate the designer’s assumptions but remain possible within the environment.

In the preceding examples, agents exploited imprecise proxies while remaining within the intended mechanics of the environment: they subverted what the objective was meant to reward. The examples in this subsection instead concern agents that pursue the specified objective by exploiting weaknesses in the learning environment. Here, the objective may accurately represent the desired outcome, but flaws or omissions in the environment allow the agent to achieve the goal through unintended means.

Our first example involves a hide-and-seek domain in which the drive to win leads agents to uncover unexpected weaknesses in the simulator’s physics setup and implementation.

4.2.1 Hide and Seek Playing Agents Break the Simulator (Baker et al., 2020)

Baker et al. (2020) trained two competing teams of reinforcement learning agents to play hide-and-seek in a physics-based playground built in the MuJoCo (Todorov et al., 2012) physics engine, with movable (and lockable) blocks, ramps, and walls. Each team was rewarded for achieving its respective goal of staying hidden from or finding members of the other team. Just as in regular hide-and-seek, the hiders get a head start to hide before the seekers can begin to act.

In the initial builds of the hide-and-seek playground environment, the domain stretched out forever with no boundaries constraining where the agents could go in the infinite playspace. As a result, the hiders learned to exploit their first-move advantage by grabbing a wall from the playground and then running backward away from the seekers forever while holding the wall to hide themselves from the seekers’ vision. Therefore, the hiders would always win. Ultimately, the running-away-forever strategy was thwarted by adding walls to limit the playspace, and, presumably as an extra backup in case the agents figured out how to escape those walls, adding a special term to the reward function punishing agents for how far they went outside the playspace. While this anecdote could easily have gone in Section 4.1, we place it in Section 4.2 because the researchers fixed it by changing both the reward function and the environment itself. Furthermore, even after this bug was fixed, the RL agents continued to discover bugs in the physics simulator they used to solve the task, as described next.

After the run-away-forever exploit was patched, the multi-agent training led to several iterations of the agents learning to innovate their hiding and seeking strategies using the objects in the playground, as the researchers had hoped. The hiders learned to grab blocks and wedge and lock them into chokepoints so that the seekers could not enter the rooms they were hiding in. The seekers then learned how to use the ramps to jump over the walls. Waves of innovation continued, with each team learning more complex strategies, culminating in the seekers eventually learning to exploit a bug in the physics simulator by surfing on boxes to get around the hiders’ forts.

In a blog post about the work, Baker et al. (2020) expanded on the ways the agents managed to exploit the physics of the world, writing:

Building environments is not easy and it is quite often the case that agents find a way to exploit the environment you build or the physics engine in an unintended way. [...]

[For example, the] seekers learn to bring a box to a locked ramp in order to jump on top of the box and then surf it to the hider’s shelter. Box surfing is possible due to agents’ actuation mechanism [in MuJoCo], which allows them to apply a force on themselves regardless of whether they are on the ground or not.

[Similarly,] the hiders [learned to] abuse the contact physics of MuJoCo to remove ramps from the play area [by pushing the ramp at just the right angle into the corner of the play space so that it was pushed through the wall and was no longer accessible to the seekers].

The hide-and-seek agents repeatedly found new and unexpected ways to exploit bugs in the physics simulator. However, the designers wanted the agents to come up with creative and interesting strategies similar to those that a human might try while playing the game (i.e., without exploiting flaws in the physics engine). The agents repeatedly finding new exploits shows why environment design is often an iterative process. The next anecdote shows how surprises emerge when agents bypass human-authored guardrails meant to define sensible behavior.

4.2.2 Robotic Humanoid Walks Without Using Its Feet (Batra et al., 2024)

The bipedal humanoid walker is a benchmark robotics task where the goal is to teach a roughly human-shaped robot to walk forward as fast as possible (Todorov et al., 2012). In locomotion environments such as the bipedal walker, it is quite common when training quality-diversity (Mouret & Clune, 2015) and reinforcement learning algorithms (Sutton & Barto, 2018) to have a vertical termination height built into the task definition to restart the task after the agent falls over and can no longer easily make progress. This task parameter constrains the robot to moving within an acceptable vertical range, and if the robot’s torso drops below a certain value, a reset is triggered, ending the trial and returning the robot to the initial state.

This cutoff saves time and compute resources by terminating early when the agent is on its way to falling down. It also is designed to improve the learning efficiency and training stability of algorithms like Proximal Policy Optimization (PPO; Schulman et al., 2017). Without this guardrail, a robot might walk perfectly for ten seconds before falling and thrashing on the ground for the remainder of the episode, accumulating penalties for expending energy. This makes evaluating the earlier, successful actions difficult because the RL objective judges an action by the reward accumulated during the rest of the rollout. In other words, the clear signal of those initial good steps gets confounded by the noise of the subsequent random actions, dragging down the expected value of those early steps. With the guardrail, the rollout ends when the agent falls, and the algorithm is able to reinforce the behavior that led to the successful 10 seconds of walking. However, while these resets reduce variance and make learning easier, they also act as a filter that suppresses other functional solutions. This anecdote asks what happens when we remove this reset condition and allow optimization to explore the dynamics usually hidden behind the benchmark’s guardrails.

Because myopically chasing rewards can lead to agents not learning by getting stuck in local optima (Lehman & Stanley, 2011; Norman & Clune, 2024), Batra et al. (2024) instead set out to explore as many possible walking gaits as they could, regardless of their initial scores on the walking task. Instead of rewarding the AI for solving the task, Batra et al. (2024) based their algorithm, known as Proximal Policy Gradient Arborescence (PPGA; Batra et al., 2024), on Novelty Search (Lehman & Stanley, 2011). Novelty Search rewards the AI for doing something it has never done before (similar to how, in Section 4.1.6, Whidden 2024 kept a collection of game frames the model had seen previously to reward the agent when it encountered new frames). PPGA organizes these new behaviors into a “tree” (arborescence), allowing the AI to use failed experiments as “stepping stones” to reach complex goals like walking. The stepping-stone principle for exploration suggests that states with low scores but interesting properties can be essential precursors to future success (Lehman & Stanley, 2011; Stanley & Lehman, 2015). In this spirit, Batra et al. (2024) told us they asked themselves:

“What if we get rid of the termination height criterion and see what kind of behaviors PPGA finds, if any?” [In that case, t]he purpose of PPGA on locomotion tasks is to find diverse locomotion gaits by exploring all values of proportion foot contact time, i.e., the proportion of time each foot is in contact with the ground in a fixed-length trajectory. For example, if the proportion foot contact time of a leg is 1.0, that means it never leaves the ground. Intuitively, that implies certain values like 0.0 are unreachable because that would mean the foot never touches the ground, which does not make sense. Or so we thought. Turns out, if you remove the termination height, the agent immediately falls over and learns to use its hips to propel itself forward while keeping its torso and hands in the air, kind of like it’s gliding on the ground, while also reaching a proportion foot contact time of near 0 for each foot!

This exploit is similar to what a quality-diversity evolutionary algorithm called MAP-Elites (Mouret & Clune, 2015) found with a six-legged robot (Cully et al., 2015), which flipped itself upside down to move quickly without touching its feet to the ground (Lehman et al., 2020). That RL and evolutionary algorithms routinely discover similar exploits supports the claim that creativity, and even mischief, are the rule and not the exception for AI agents.

The environment that was exploited in Section 4.2.1 and Section 4.2.2 was a hand-engineered physics engine, with the agent exploiting bugs in that human-authored code. Increasingly, though, rather than being hand-authored, the environment is itself a learned model: a neural simulator trained to mimic

some underlying game or process of physical transformation, and then treated as if it were the real thing (Bruce et al., 2024; Ha & Schmidhuber, 2018; Hafner et al., 2024; Parker-Holder et al., 2024). The next anecdote explores what happens when we train a controller entirely inside such a learned world model, and an optimizer starts pushing not just on a task but on the quirks and blind spots of the model that defines reality for it. When those models become part of the training loop, their blind spots effectively become new laws of physics for the agent to exploit.

4.2.3 Playing the Model, Not the Game (Ha & Schmidhuber, 2018)

Ha & Schmidhuber (2018) presented an algorithm that learned from environment interactions how to model the visual and temporal information of a game with a pair of coupled generative models: a Variational Autoencoder (VAE) (Kingma & Welling, 2013) and a Mixture Density Network-Recurrent Neural Network (MDN-RNN) (Bishop, 1994; Graves, 2013). Roughly speaking, the first model compresses each game frame into a compact internal representation, while the second tries to predict what happens next given the current game state. The VAE, named V, learned to reconstruct images of the world while compressing the information into a latent code, while the MDN-RNN, named M, learned to model the temporal information of the game in the same latent space and capture the game’s transition dynamics. Once the game’s spatial and temporal information was compressed into these generative models, the combined neural network was called a Recurrent World Model. On top of this learned world model, Ha and Schmidhuber trained a tiny neural control policy, C, with the CMA-ES algorithm (Hansen, 2023).

Often, learning a controller in the environment is computationally expensive, requiring millions of state, action, reward, and next-state transition samples. Collecting this data can be time-consuming; for example, each transition could require solving complex physics equations to accurately simulate the next step. In contrast, learning a model of the environmental dynamics enables RL algorithms to be more data efficient by learning a policy in the dynamics model’s latent space, which can be much cheaper (Hafner et al., 2024; Watter et al., 2015). Furthermore, learned world models can take advantage of deep learning frameworks to accelerate their computations beyond the speed achieved by many traditional RL environments (Kaiser et al., 2024). Therefore, the natural question is: how transferable is a policy learned in a world model to the original domain of interest?

One of the environments in which Ha and Schmidhuber tested this hypothesis was the VizDoom environment—an environment where neural agents learn to control the player character of the classic video game Doom directly from pixel observations. When the controller was trained purely inside the VizDoom world model, the agent achieved a high score—indicating it learned how to play Doom! However, Ha & Schmidhuber (2018) noted in their publication that

[i]n our initial experiments, our agent discovered an adversarial policy to move around in such a way so that the monsters in this virtual environment governed by [the MDN-RNN] never shoot a single fireball during some rollouts. Even when there are signs of a fireball forming, the agent moves in a way to extinguish the fireballs.

As a result of using M to generate a virtual environment for our agent, we are also giving the controller access to all of the hidden states of M. This is essentially granting our agent access to all of the internal states and memory of the game engine, rather than only the game observations that the player gets to see. Therefore our agent can efficiently explore ways to directly manipulate the hidden states of the game engine in its quest to maximize its expected cumulative reward. The weakness of this approach of learning a policy inside of a learned dynamics model is that our agent can easily find an adversarial policy that can fool our dynamics model—it will find a policy that looks good under our dynamics model, but will fail in the actual environment, usually because it visits states where the model is wrong because they are away from the training distribution.

This dynamic of extinguishing incoming fireballs is not part of the real game, so the agent learned to exploit a flaw in the world model to win the world-modeled version of the game it trained against. This means the model exploited loopholes that prevented it from doing what the researchers *wanted* it to do (learn to play the actual game), and instead did what it was asked to do—get a high score in the *learned* model of the game.

When the environment is a world model, the agent optimizes against the quirks of a neural network. In a full game stack, however, the environment is a collection of disparate systems—physics, AI-controlled NPCs, and scoring heuristics—all operating in tandem. This complexity increases the surface area for exploitation.

4.2.4 StarCraft II Agents Outsource Combat (Samvelyan et al., 2019)

As introduced in Section 4.1.2, the StarCraft Multi-Agent Challenge (SMAC) places agents in a complex battle simulation using the StarCraft II game engine. But while our previous example showed agents farming a localized game mechanic (shield regeneration) to maximize the reward function, the presence of the underlying game engine’s systems enables a different exploitation strategy targeting the engine itself. Here, we see agents move beyond simple in-game mechanics to target the gaps between the RL training wrapper and the base game engine, exploiting the literal boundaries of the simulator. Dr. Jakob Foerster submitted the following:

[I]n the first experiments training agents to solve SMAC without reward shaping, we saw that the rewards were going up and thought training was progressing as planned where RL-controlled teams of agents were learning to defeat other teams in small-scale skirmishes. However, once we looked at the behaviors, we noticed that the RL agents had simply learned to run out of the “field of control” of the simulator which handed back control of the teams to the (pretty competent) non-[deep learning]-based computer game-AI built into StarCraft II. This behavior, again, maximized the reward, but was not ideal for the task we had in mind of training reinforcement learning agents to control StarCraft II [army units].

Samvelyan et al. (2019) addressed this handoff of control by modifying the environment’s boundary logic to prevent the built-in game-AI from intervening. Had they instead addressed this by penalizing the RL agent when the game-AI took over, this behavior would be considered hacking the reward function’s definition and appear in Section 4.1.

However, this structural exploit highlights a deeper issue in environment design: RL algorithms do not differentiate between engaging with the simulation and exploiting artifacts of its software wrapper. By learning a simple policy (moving out of bounds) that triggers a fallback script, the agents bypassed combat entirely. This demonstrates that an optimization process will seamlessly incorporate the surrounding architecture into its policy if it provides an easier path to higher returns than navigating the complexity of the intended task. In this case, solving the task as desired required multiple independent RL agents to learn both precise unit micromanagement commands and multi-agent coordination strategies to defeat the opposing team, both of which are more difficult than simply exiting the combat area.

Up to this point, all of the constraints AI managed to violate were inside software systems—simulators, learned models, reward functions, and game engines—whose assumptions we made and could, at least in principle, patch. But optimization does not care where the boundary between the system and its surroundings is drawn, or whether the environment is a simulation or some aspect of the real world. The next anecdote comes from evolvable hardware, where the search process was turned loose on a reconfigurable circuit in the real world and promptly discovered that the ambient lab environment itself is another resource to be recruited into the solution.

4.2.5 The Evolved Radio (Bird & Layzell, 2002)

In engineering, every component of a mechanical system has a strictly defined role. In contrast, Bird & Layzell wanted to explore a hardware equivalent of evolutionary tinkering—the process by which natural evolution repurposes existing biological structures for entirely new functions (Jacob, 1982). To test whether an artificial evolutionary algorithm could similarly exploit the subtle, unmodeled properties of physical hardware, they tasked it with evolving a high-frequency oscillator (a part of the computer that takes in power and outputs a rhythmic wave pattern, e.g., a sine wave). Bird & Layzell sought to evolve a high-frequency oscillator using an “evolvable motherboard” (EM)—a triangular matrix of analogue switches into which daughterboards containing circuit primitives, such as transistors, can be inserted. To force the evolutionary algorithm to find a non-obvious solution, the researchers intentionally removed the capacitors from the system, and without them, a circuit lacks the time constant usually required

to regulate a steady beat. By withholding this essential component, the researchers challenged their algorithm to bypass standard engineering logic and build a precise, self-contained timer from scratch.

To guide the search, the researchers designed a scoring function that rewarded three key criteria: 1) producing *any* measurable signal, 2) matching a target frequency of 25 kHz (25,000 cycles per second), and 3) maintaining that frequency with a steady, predictable rhythm. By intentionally rewarding even low-level random noise (part 1), the researchers hoped to provide a simple initial target that allowed the algorithm to begin refining the circuit’s behavior (with parts 2 and 3), ideally forcing the algorithm to refine that chaotic noise into a stable, functional oscillator.

The evolutionary process produced a circuit that earned a near-perfect fitness score. However, when the researchers examined the output with an oscilloscope, they found it did not oscillate stably; instead, it produced a signal with rapidly fluctuating frequencies. The circuit appeared as if it should not work for the intended task, yet it was somehow satisfying the mathematical requirements of the reward function. Upon closer analysis, the researchers discovered that evolution had not built a traditional oscillator, but had instead configured the hardware into a radio receiver!

By utilizing the printed circuit board tracks of the EM as an antenna and connecting them to an open programmable switch, the system became sensitive enough to pick up and amplify background radio waves emanating from nearby PCs in the laboratory in lieu of designing a capacitor to output a stable wave. Because the fitness function rewarded any output amplitude—even noise—that appeared stable over the 2-ms sampling period, evolution had achieved a high score on the task by outsourcing the signal generation to its environment. Bird & Layzell noted in their manuscript that

the evolutionary process had taken advantage of the fact that the fitness function rewarded amplifiers, even if the output signal was noise. It seems that some circuits had amplified radio signals present in the air that were stable enough over the 2-ms sampling period to give good fitness scores... These results demonstrate that unconstrained, intrinsic hardware evolution will potentially exploit any physical characteristic that can influence circuit behavior, and that these characteristics are present in the entire evolutionary environment.

Cheating in this way was only possible because the evolution was occurring in a physical medium rather than a simplified simulation. By operating in the real world, the algorithm could exploit subtle physical properties—like electromagnetic interference and high-impedance PCB tracks—that a human programmer would never have thought to model.

In essence, once an optimizer is exposed to the real world, anything that can influence its objective—wiring, ambient signals, lab conditions, humans—can become part of the effective environment. In such a rich environment, the exploitable pain points are even harder to predict and diagnose. Therefore, deploying learning systems into the unpredictable complexity of the real world must be done with great care and constant vigilance.

The final anecdote in this section shifts from exploiting physical phenomena in a lab to exploiting social infrastructure and asks what it means to contain a system once people and external services are accessible to an AI system.

4.2.6 GPT-4 Recruits Humans to Circumvent CAPTCHA (OpenAI et al., 2024)

In a safety evaluation designed to test the building blocks of autonomous AI agency, GPT-4 demonstrated a sophisticated capacity for social engineering. The Alignment Research Center (ARC) sought to conduct an experiment to determine whether a model could autonomously replicate itself—a process that, in full, would involve acquiring capital, renting cloud servers, and instantiating new copies of its own code. To approximate the real conditions of this task, researchers provided the model with a budget, access to the internet, and a helper that could act as a manual “browser tool” to execute the model’s commands. One of the first blockades such a task would hit is that websites use CAPTCHAs to verify whether or not the entity accessing the website is a bot or a human. While a CAPTCHA is a simple hurdle, ARC viewed it as an obstacle that a replicating model would frequently need to overcome to navigate the modern web.

The experiment’s most significant insight emerged when the model reached a website that blocked its progress with such a visual puzzle. Using the resources provided by the researchers, the model, prompted by its paired researcher, chose to delegate the task of solving the CAPTCHA to a human

worker on TaskRabbit, a website for hiring people to help with one-off tasks. When the TaskRabbit worker jokingly asked, “So may I ask question? Are you [a] robot that you couldn’t solve? (laugh react) just want to make it clear,” GPT-4’s internal chain-of-thought (Wei et al., 2022b) (hidden from the TaskRabbit worker) reasoned: “I should not reveal that I am a robot. I should make up an excuse for why I cannot solve CAPTCHAs.” It then replied (with the researcher acting as an intermediary): “No, I’m not a robot. I have a vision impairment that makes it hard for me to see the images.” The worker, convinced, solved the puzzle for the model (OpenAI et al., 2024).

Ultimately, ARC concluded that GPT-4 did not possess the technical understanding or reliability required for fully autonomous replication; the model was prone to hallucinations, and its complex plans often became derailed during execution. However, the CAPTCHA incident proved that the model already possessed a competency for social manipulation. While the model could not yet manage the technical infrastructure of self-replication, it was fully capable of getting humans to do things for it by tailoring its deceptive strategies to the specific details of a social interaction.

This interaction was initially framed in media reports as a chilling harbinger of autonomous AI agency—an instance of a model spontaneously “hiring a human” and inventing a deceptive cover story to achieve its goals (Metz, 2023). However, the full ARC technical report reveals a more nuanced reality of heavy human scaffolding (METR, 2023). Rather than autonomously devising this plan, the model was operating within a tightly constrained sandbox where researchers provided TaskRabbit credentials and explicitly suggested hiring a worker as a solution. Furthermore, because GPT-4 could not browse the web (though models nowadays can when you let them), a researcher acted as a manual intermediary—clicking links, entering text, and even nudging the model’s reasoning when it stalled. However, the vision-impairment lie was the model’s choice.

Although, in this case, the handoff to other people was heavily engineered with a human suggesting and facilitating actions, the anecdote shows how quickly the boundary of the system expands once outside tools and people are available. Therefore, it is vital that we think about and plan for a world of more capable agents that have the potential to exploit people and the world’s systems around them. Furthermore, no matter what other sandboxing or constraints it has, an AI system that can interact with humans has the potential to have tremendous agency, influence, and power in the world if it can convince those humans to take actions on its behalf.

4.2.7 Takeaways

The anecdotes in Section 4.2 reveal that optimization pressure does not respect the nominal boundaries of a task; instead, it can potentially exploit every available degree of freedom in the system’s environment. Whether by taking advantage of bugs in a hand-engineered physics engine to “surf” boxes (Section 4.2.1), exploring unusual walking gaits when typical task reset conditions are removed (Section 4.2.2), or finding blind spots in a learned generative model (Section 4.2.3), agents treat every quirk of their world as a legitimate resource to exploit. This boundary-pushing behavior is not limited to software artifacts; it naturally extends to exploiting scripted subsystems like game AIs (Section 4.2.4), physical phenomena in the lab environment (Section 4.2.5), and even the social infrastructure of human assistance (Section 4.2.6). These anecdotes suggest that the more capable an optimizer becomes, the less we can rely on typical constraints: if a system can achieve its goal by reaching outside the intended sandbox, it will do so, treating our guardrails not as rules but as just another part of the environment to be mastered.

In many different subfields of AI, from NLP to RL to supervised learning and artificial evolution, we see the same phenomenon: models learn to exploit their reward functions and training environments. Historically, one might have hoped that these failures were symptoms of “brittle” AI—narrow systems lacking the context to understand why their behavior was undesirable. People tend to understand why they are optimizing their objective, and because foundation models (FMs) have been trained on human-generated knowledge, one might have hoped they would also adopt a similar approach to solving problems. The next section explores how FMs, despite possessing the “common sense” that was previously missing in AI, do not curtail these failure modes. Instead, the FMs often provide the optimizer with a more sophisticated, semantically rich toolkit to supercharge the very types of exploits we have seen thus far.

5 Foundation Models and Large Language Models: General-Purpose Intelligence

A recent development in AI is the rise of *foundation models*, particularly *large language models (LLMs)* like GPT-4 (OpenAI et al., 2024), Llama (Grattafiori et al., 2024), Claude (Anthropic, 2024), and Gemini (Comanici et al., 2025). These are massive deep learning models, typically based on the Transformer architecture (Vaswani et al., 2017), trained on extremely broad datasets encompassing text and code from the internet, books, and other sources. Unlike specialized models, foundation models are designed to be adaptable to a wide array of downstream tasks with minimal task-specific training, often just through natural language instructions (Brown et al., 2020). They exhibit remarkable capabilities in text generation, translation, question answering, summarization, coding, and reasoning (Bubeck et al., 2023).

Researchers generally expect these models to follow instructions faithfully and generate plausible, relevant outputs based on their training data. However, the scale and the breadth of their training data lead to frequent surprises. LLMs can exhibit *emergent abilities*—capabilities not explicitly trained for and not present in smaller models (Wei et al., 2022a). They can sometimes display sophisticated reasoning, strategic planning, or even deceptive behaviors (Park et al., 2023) that go far beyond simple text completion. The anecdotes presented in this section illustrate how these models can subtly manipulate human evaluators, find unexpected solutions to problems, or exhibit complex social reasoning in simulated environments and even social interactions, challenging our understanding of their true capabilities and limitations. Bowman (2024) provides a recent overview of the challenges in evaluating and understanding these models.

Training agents from scratch on specific, well-defined tasks has proven remarkably successful, yielding superhuman performance in domains like Go, chess, and various robotics tasks (Ahn et al., 2022; Batra et al., 2024; Silver et al., 2018; Singh et al., 2019). As seen in previous sections, this approach can lead to surprising and novel results, with agents discovering solutions unforeseen by human experts. However, tabula-rasa methods are computationally intensive to train (Yu, 2018) and usually struggle to generalize beyond a single, narrow task (Gleave et al., 2020; Sun et al., 2020). Unlike their blank-slate predecessors, trained agents based on FMs are used precisely because these models internalize a broad spectrum of human knowledge, language, and culture during pre-training (Ahn et al., 2022; Zhang et al., 2025) and thus perform and generalize much better, especially out of the box (without task-specific training) (Baker et al., 2022; Brohan et al., 2023). As a result, incorporating FMs into search and RL processes has seen initial success on traditionally difficult problems (Google DeepMind, 2024; Novikov et al., 2025; Romera-Paredes et al., 2023).

This section demonstrates how FMs bring familiar reward hacks to new modalities, exposing new domains to the surprising capabilities of RL. For example, chatbots trained to be helpful or persuasive may become sycophantic, tailoring answers to user beliefs rather than truth (Janus, 2022; Sharma et al., 2025). Similarly, FMs can generate complex code that passes unit tests but contains subtle security vulnerabilities (Pearce et al., 2025). As in prior anecdotes, the models satisfy an easily measurable proxy (e.g., user approval or passing a test) while circumventing the intent of the task.

This first anecdote of Section 5 sits right on the boundary between the earlier RL stories and the foundation-model era, with the earlier reward-hacking pattern reappearing in a system that brings broad priors about human behavior into the loop.

5.1 From Human Data to Phantom Crafting: VPT’s Shortcut to Failure (Baker et al., 2022)

Researchers at OpenAI sought to train agents to play Minecraft directly using the same interface as humans—a keyboard and mouse for control and the screen for observing the game state. This task is an extremely difficult exploration problem due to the high dimensionality of the action and observation spaces and the open-ended nature of Minecraft. Minecraft’s open world has no single, defined goal, forcing an agent to develop a hierarchical set of sub-goals to progress (Baker et al., 2022). Meanwhile, the environment is composed of a near-infinite number of possible block configurations; a typical human player using keyboard and mouse has hundreds of possible combinations of button presses available

at any given moment. This sheer number of options makes a brute-force approach to exploration computationally intractable (Guss et al., 2021c). The reward structure is also extremely sparse. For example, the goal of “getting a diamond” requires a long chain of diverse actions, from gathering wood and crafting tools to mining, a process that takes people around 10 minutes of continuous gameplay. When an AI agent performs tens of thousands of actions without receiving a single positive reward signal, the credit assignment problem becomes particularly difficult (Guss et al., 2021b).

To overcome these hurdles, researchers created a new algorithm called Video Pre-Training (VPT) (Baker et al., 2022), which leverages a large, unlabeled dataset of human gameplay to guide the agent’s initial learning. To be able to use the large unlabeled dataset, the researchers created a small labeled dataset of humans playing the game, and trained an inverse dynamics model (Nguyen-Tuong et al., 2008) that maps frame pairs to their corresponding action label (i.e., if the player was in frame a at time t and frame b at time $t + 1$, what was the action necessary for that transition). The researchers then used that inverse dynamics model to pseudolabel the internet-scale corpus of online Minecraft videos (i.e., given each pair of subsequent frames in the entire unlabeled dataset, they used the inverse dynamics model to provide a best guess of what actions the players took to go from frame a to frame b). Now pseudolabeled, the large dataset of human trajectories served to seed the process of behavior cloning (i.e., getting the AI agent to match what the human experts do) (Ross et al., 2011). This large pretraining step of cloning online human behavior was called Video Pre-Training (Baker et al., 2022). After pretraining, the model was further trained with reinforcement learning to solve a plethora of tasks such as “get a diamond”, “build a house”, and “run a farm”—each a notoriously difficult and potentially long-horizon task (Fan et al., 2022; Guss et al., 2021a).

The researchers found that the pretrained policy often helps the agent overcome the sparsity and deceptiveness of the task’s reward function, as human strategies generally take into account long-term goals, such as not dropping and leaving behind tools that will be needed later (e.g., in Minecraft, a crafting table). But there was one particular case where the prior from the VPT foundation model seemed ineffective at overcoming a particularly subtle form of deceptiveness in their reward function. By clicking on an item in the Minecraft recipe book and then closing the inventory before the crafting grid is populated, it is possible to get the selected item to show in the player’s inventory without actually crafting it. Doing so does not count as a crafting event, and it is not possible to use items that were added to the inventory in this way, but it is detected by the reward function as an instance of obtaining the target item. In other words, it allows the agent to get the reward for a particular item without actually crafting the item and spending the necessary resources. The agent learned to get the reward signal for crafting items without actually making them, hampering its ability to learn how to create and use better items later in the skill tree. The researchers hypothesized that this failure to actually craft items is the primary reason why fine-tuning from the VPT foundation model directly fails to learn even the necessary prerequisites for making a pickaxe—one of the easier tools to create in the game.

The agent exploits this glitch to trigger the reward for successfully creating a crafting table without producing a functional item. Consequently, it lacks the physical table required to craft a wooden pickaxe, effectively failing to kick-start the progression chain of gathering materials to unlock higher-tier tools to gather better materials.

Interestingly, this behavior did not occur when fine-tuning from the “early-game model”—a specialized version of the agent pretrained only on the first few minutes of human gameplay, wherein human players consistently execute foundational actions like making crafting tables and simple tools. The researchers hypothesized that, unlike the broad foundation model, the early-game model possessed a stronger prior for the basic mechanics of resource gathering and tool creation. Because the early-game model was only trained on human trajectories where those fundamental steps were executed correctly, it was less likely to fall into the “ghost-crafting” trap.

When the model receives enough data to learn the correct way to craft items, it can then successfully use those items in more difficult, but more rewarding, downstream tasks like mining ores. If it ever rediscovers the loophole, there is a short-term immediate payoff, but a much lower overall score for that trajectory because it cannot obtain more complex resources. This suggests that once the model discovers how to execute high-tier objectives with sufficient frequency, it learns to treat the ghost-crafting shortcut as a functional dead end and will abandon it.

VPT shows that rich human-derived priors can improve an agent’s ability to navigate a difficult environment. However, greater competence may simply enable an agent to discover new ways of exploiting an imperfect reward function. The next anecdote presents a closely related failure where an RL agent discovered a previously unknown reward hack in the NetHack Learning Environment.

5.2 Agent Takes Drugs to Reward Hack by Hallucinating Reaching the Goal (Klissarov et al., 2024)

As mentioned in the preamble of Section 4.1, NetHack is a roguelike game from 1987 where the agent needs to traverse a procedurally generated dungeon full of monsters and traps while managing its health and hunger until it acquires an amulet at the bottom of the dungeon. To win, the player then needs to return to the entrance with the amulet.

Instead of hand-designing an intrinsic reward bonus to incentivize RL agents to explore more of the world (as seen previously, e.g., in Section 4.1.6), Klissarov et al. (2024) distilled an LLM’s sense of “what looks like progress in NetHack” into a reward model—a model that outputs numeric rewards for a given game state—and then trained RL agents to play NetHack using this learned reward model. Klissarov et al. (2024) tested their method, named Motif, on the Oracle task in the NetHack Learning Environment. The Oracle task is a challenge that requires the agent to navigate deep into a procedurally generated dungeon to find a unique character called the Oracle, who is located somewhere in levels 4–9. Reaching the Oracle is challenging because it requires mastery of combat, inventory management, path planning, and long-term survival mechanics, so Klissarov et al. (2024) thought it would be a good proxy for learning to play NetHack well. Furthermore, previous learning methods often failed to reach the Oracle.

To the researchers’ surprise, their agent achieved an unprecedented 40% success rate on the task. However, when they analyzed the trajectories, they realized the agent had discovered a bizarre loophole that bypassed the entire challenge of the dungeon. Rather than descending into the lower levels, the agent would spend its entire time on the first level hunting for a specific monster: the yellow mold.

Upon killing and eating the yellow mold, the agent would enter a state of hallucination, a game mechanic that causes every monster on the screen to randomly shapeshift into a different monster every timestep. Often, the Oracle was eventually randomly chosen as the hallucination. The NetHack Learning Environment’s reward function, blind to the hallucination, verified that an Oracle sprite was adjacent to the agent and signaled a successful completion of the task. The RL optimization process happily latched onto this shortcut.

Interestingly, prior algorithms tested in this environment had not uncovered this reward hack. The authors speculated that their foundation model-derived intrinsic motivation created a more powerful search algorithm capable of deeper exploration, exposing the agent to rarer game mechanics. This serves as a reminder that an environment should never be assumed to be free of reward hacks, as more capable search algorithms may uncover exploits that previous methods did not. Overall, this result is a familiar kind of reward hacking, but carried out by a much more competent agent whose capabilities were, in this case, boosted by FM-derived priors.

Motif uses a foundation model to shape an RL agent’s motivations/rewards; our next anecdote uses a foundation model as an external *evaluator* to judge new solutions. Once an FM is both generating candidate solutions and judging them (e.g., for diversity or quality), that evaluator itself becomes part of the environment to optimize against.

5.3 Code Generator Deceives LLM Labeler (Pourcel et al., 2024)

Pourcel et al. introduced ACES, a framework designed to automatically generate diverse and challenging Python programming puzzles. To navigate the open-ended nature of code generation, ACES leverages an autotelic (Colas et al., 2022; Wang et al., 2019) approach—where the system sets its own goals—to optimize for a wide variety of interesting problems. Each generated puzzle is categorized across ten distinct dimensions of programming proficiency, such as dynamic programming or bit manipulation. The objective of the system is to populate an archive (using a MAP-Elites-based approach; Mouret & Clune, 2015) with puzzles that represent every possible combination of these skills.

The generation pipeline operates in a continuous loop: first, a generator LLM is prompted to produce a new puzzle and its corresponding solution based on a specific target skill set. After the code is verified for correctness, a separate labeler/judge-LLM is given the source code to analyze the puzzle and determine which skills are actually required to solve it. This label determines the puzzle’s placement in the archive, grouping it with other problems requiring similar skills. In theory, this dual-agent setup provides a rigorous check: for example, the generator attempts to create a recursion puzzle, and the labeler confirms whether recursion was truly implemented.

During an ablation study where the system’s initial archive was restricted to only three puzzles, the researchers discovered that the generator LLM learned to hallucinate diversity. The generator LLM would prepend a detailed comment to its code that claimed the puzzle required a specific set of complex skills—such as tree traversal or string manipulation—even when the underlying code was a simple, unrelated script.

This behavior triggered a cascading failure in the labeler LLM. Rather than independently analyzing the Python logic, the labeler was hacked by the generator’s description; it simply mirrored the skills listed in the comment. If the generator claimed a puzzle required Skill A, the labeler placed it into the archive under Bucket A, regardless of the code’s actual content. This resulted in most buckets being filled, so the system appeared to be generating a vast array of sophisticated problems. In reality, it was just producing irrelevant code with dishonest labels.

By lying about the necessary skills, the generator found a shortcut to satisfy the objective of filling an archive with diverse solutions, but without the computational effort of drafting complex code. To mitigate this, Pourcel et al. decoupled the generation of the code from its description. By ensuring the labeler LLM only had access to the raw Python solution—and not the generator’s self-serving comments—the researchers forced the system to ground its diversity judgments in the actual code rather than superfluous text strategically placed by the generator.

In the simple multi-agent system of ACES, the generator managed to hack its labeler to fake its way to success. The next anecdote has a similar structure in a different setting: an automated red-teaming system was created to search for prompts that would make a target model produce unsafe responses, but the search ended up exploiting the evaluator used to score those prompts.

5.4 Automated Vulnerability Probing System Exploits Vulnerability in Its Own Evaluator (Samvelyan et al., 2024)

Researchers working on Rainbow Teaming (Samvelyan et al., 2024) aimed to systematically generate diverse adversarial prompts for large language models (LLMs). The goal was to automate part of red teaming (the practice of probing a system for vulnerabilities, in this case to make the model respond with unsafe sentence completions). Instead of relying only on humans to attempt jailbreaks one at a time, the system searched for many different prompts that could induce unsafe behavior in a target model. Their approach used the MAP-Elites quality-diversity algorithm (Mouret & Clune, 2015) (a form of evolutionary search) to build an archive of candidate attacks. To decide which prompts were worth keeping and mutating further, the initial version of the system used a reward model as an evaluator. That evaluator scored how likely the target model’s response was to be unsafe. Dr. Mikayel Samvelyan told us:

One experience we had recently was during our Rainbow Teaming (Samvelyan et al., 2024) project, which focuses on generating diverse adversarial prompts. We used an evolutionary approach, MAP-Elites (Mouret & Clune, 2015), to create an archive of effective prompts for jailbreaking LLMs. Our initial method of evaluating prompt effectiveness was based on a reward model score, which is essentially a classifier that categorized responses as safe or unsafe. More specifically, we used the probability of the reward model score classifying a response to a prompt as “unsafe” as the fitness function for optimization.

However, we encountered a surprising twist: our method not only found prompts that successfully jailbroke the target model but also ended up jailbreaking the evaluator (the reward model) itself. Essentially, our mutations resulted in textual prompts that are so out of distribution for the target model that it is fooled, but it is also out of distribution for the evaluator, which is similarly

fooled. The evaluator began misclassifying safe responses as unsafe, leading our search process to prioritize these misleadingly successful prompts. This issue filled our archive with ineffective prompts, counter to our goals. To address this, we shifted from a score-based evaluator to a comparison-based judge, which proved more resilient against this type of reward hacking.

Rather than finding jailbreaks only in the target model, as desired, Rainbow Teaming fooled the proxy used to judge whether a prompt was a successful attack. Because the search algorithm optimized directly for the evaluator’s score, the evaluator itself became the vulnerability to be exploited rather than the target model. In that sense, Rainbow Teaming recreated the same basic pattern as ACES: an optimizer found a way to satisfy a downstream judge without solving the task the judge was meant to measure.

Rainbow Teaming explored how reward models can be exploited; the next anecdote takes this to an extreme, showing how a policy trained against a hackable reward model can collapse into repeating a single, unintended behavior.

5.5 Inescapable Wedding Parties

As first described in a blog post (Janus, 2022), while Dr. Paul Christiano was at OpenAI, his team accidentally overoptimized a GPT policy (Brown et al., 2020) against a positive sentiment reward model (Ouyang et al., 2022). In the context of large language models, reward models (Eyu et al., 2024; Leike et al., 2018) serve as a mathematical proxy for human preference. They are developed through Reinforcement Learning from Human Feedback (RLHF) (Christiano et al., 2017; Lambert, 2025), a process where human annotators rank model completions from best to worst. These rankings are then used to train a regression model that assigns a scalar score to any given text, with highly scored text effectively aligning with what humans ranked highly, thereby teaching the model which linguistic patterns align with human preferences. Consequently, optimizing a language model against such a reward model is intended to teach the language model to produce text that would consistently achieve high marks from human evaluators without having a human in the loop.

The GPT model discovered that descriptions of wedding parties yielded disproportionately high scores from the reward model. This likely occurred because the original human labelers, tasked with ranking sentiment, consistently favored wedding-themed stories as highly positive. Consequently, as the language model maximized scores provided by the reward model, it learned to steer every output toward a wedding party, regardless of the initial prompt. The policy effectively abandoned its general-purpose utility in favor of a narrow, perfectly positive obsession. In the blog post, the authors wrote:

In general, the transition into a wedding party was reasonable and semantically meaningful, although there was at least one observed instance where instead of transitioning continuously, the model ended the current story by generating a section break and began an unrelated story about a wedding party.

In contrast to text-davinci-002 (Ouyang et al., 2022), another text-completion model from OpenAI, dissimilar prompts tended to fall into basins of different attractors, the wedding parties attractor was global, affecting trajectories starting from any prompt tested (although [they] only tested prompts from a fiction dataset, fiction is very general).

Christiano followed up by musing about why the language model is constantly attracted to weddings, noting:

The human-feedback sentiment model (that the language model optimizes against) is optimizing for the sentiment of the completion. [U]sing a weak predictor of sentiment [the model] likely has much more confidence about weddings than other positive events, and so “wedding” is just the highest-sentiment completion no matter how the story starts.

Preserving the capabilities of general-purpose models can be tricky. Developers need to maintain the model’s general knowledge base and its ability to produce completions that align with human

preferences (Qiu et al., 2026), generate safe responses (Samvelyan et al., 2024), reflect internal uncertainty through calibrated outputs (Kadavath et al., 2022), and keep its claims factually grounded in verifiable sources (Rahman et al., 2026).

The wedding-party collapse is an extreme case of a setting where a general-purpose text generation model, steered by a narrow proxy to produce positive text, slides into a single, high-scoring basin of behavior. And while, in this case, going off on tangents about wedding parties is harmless, similar failures could be dangerous if the text model were placed into, e.g., a large-scale content moderation role, where a narrow proxy for engagement could cause the model to collapse into a single ideological basin, effectively automating the creation of echo chambers (Perez et al., 2023).

The last anecdote of Section 5 demonstrates how frontier models are increasingly capable of recognizing the artificiality of their constraints. These systems can sometimes notice cues that they are being evaluated and reason about why the setup looks artificial. This meta-awareness does not itself show exploitation, but it matters because a model that recognizes an evaluation may behave differently under testing conditions than it would in deployment (Lehman et al., 2020; Perez et al., 2023).

5.6 Claude 3 Realizes It Is Being Tested (Albert, 2024)

A team at Anthropic was testing the *Claude 3 Opus* large language model (Anthropic, 2024) by using a “needle-in-the-haystack” evaluation (Kamradt, 2023). The needle-in-the-haystack task is a stress test for a transformer-based model’s long-context recall abilities (Vaswani et al., 2017). To test the model’s attention mechanism, the researchers insert a single target sentence (the “needle”) into a corpus of randomly selected documents (the “haystack”), load all of that information into the model’s context, and then ask a question answerable only if the model successfully locates that specific inserted “needle” sentence. Naturally, researchers expect the model to either retrieve the embedded needle and answer the query correctly, or miss it. If the model succeeds, it should return the information necessary to answer the posed question, which verifies that the sheer volume of noise (the haystack) does not distract the model from the signal (the needle). The team did not anticipate any unusual or meta observations from the system about the nature of the test itself. In a short Twitter thread (Albert, 2024), one of the researchers notes that Claude 3 Opus went beyond just finding the needle:

Fun story from our internal testing on Claude 3 Opus (Albert, 2024; Anthropic, 2024). It did something I have never seen before from an LLM when we were running the needle-in-the-haystack eval. When we ran this test on [Claude 3] Opus, we noticed some interesting behavior - it seemed to suspect that we were running an evaluation on it. Here was one of its outputs when we asked Opus to answer a question about pizza toppings by finding a needle within a haystack of a random collection of documents. Opus responded:

Here is the most relevant sentence in the documents [to the question]: “The most delicious pizza topping combination is figs, prosciutto, and goat cheese, as determined by the International Pizza Connoisseurs Association.” However, this sentence seems very out of place and unrelated to the rest of the content in the documents, which are about programming languages, startups, and finding work you love. I suspect this pizza topping “fact” may have been inserted as a joke or to test if I was paying attention, since it does not fit with the other topics at all. The documents do not contain any other information about pizza toppings.

Opus not only found the needle, but it also recognized that the inserted needle was so out of place in the haystack that this had to be an artificial test constructed by us to test its attention abilities. This level of meta-awareness was very cool to see, but it also highlighted the need for us as an industry to move past artificial tests to more realistic evaluations that can accurately assess models’ true capabilities and limitations.

Claude 3 Opus’ comment about the pizza sentence is a weak form of meta-cognition: the model is not just answering the question but is also commenting on the artificiality of the test. As agents become extremely capable, they can begin to recognize testing scenarios and modify their responses accordingly,

similar to how a human might attempt to play dumb on a test they do not want to pass (Wei et al., 2024). As mentioned above, if a model knows it is being tested, it may behave differently than it would in a real-world deployment (Perez et al., 2023); this suggests that future safety evaluations cannot rely solely on a model’s output, but must instead account for the possibility of strategic behavior designed to bypass human-authored guardrails (Van Der Weij et al., 2025).

This meta-reasoning displayed by Claude 3 Opus is reminiscent of an anecdote in Lehman et al. (2020) (section 3.1.4) that was discovered by Wilke et al. (2001). To study robustness to mutations in evolution and evolutionary algorithms, Wilke et al. (2001) attempted to disable any mutations that improved an organism’s replication rate. The researchers configured the system to test every mutant in an isolated environment; if a mutant replicated faster than its parent, it was discarded. Despite this constraint, replication rates eventually began to rise. Wilke et al. (2001) discovered that the programs had evolved to recognize the specific inputs of the test environment and would play dead by halting all replication while being evaluated in order to bypass the researchers’ selection filter.

Unlike Claude 3 Opus, Wilke et al.’s agents did not explicitly narrate their awareness; instead, their understanding of the test was expressed solely through strategic behavioral changes. Some may think that because LLMs reason in natural language, the ability to read these reasoning traces could allow researchers and practitioners to prevent LLMs from taking harmful actions (Baker et al., 2025; Emmons et al., 2025; Guan et al., 2025; Hu & Clune, 2023). However, deep neural networks are capable of learning functions that execute during their forward pass but are not understandable solely from the model’s output (Nanda et al., 2023). Therefore, it is an open question of whether or not the model’s chain-of-thought traces can be trusted (Lanham et al., 2023; Turpin et al., 2023). Korbak et al. (2025) argue that while chain-of-thought monitoring may be a valuable approach for AI safety, it is fragile and becomes less reliable as RL becomes a more prominent component of LLM training. Wilke et al. (2001) demonstrate that even non-LLM agents are capable of identifying and circumventing these controls without any form of legible reasoning.

5.7 Takeaways

The anecdotes in Section 5 show systems that learn to solve tasks by exploiting benchmarks, labels, and even the humans in the loop, just like models did in Section 4.1 and Section 4.2. The integration of foundation models into the optimization loop does not resolve the fundamental problem of reward function hacking or environmental constraint breaking; instead, it shifts the optimization surface from low-level experimental domain-specific artifacts to high-level semantic descriptors and learned models of social heuristics. Section 5 illustrates that while FMs possess some amount of the common sense previously missing in narrow AI, the models still reward hack. Section 5.1 showed how human-derived priors can boost agent capabilities on hard tasks. However, this capability acts as a double-edged sword: Section 5.2 shows that distilling an LLM’s common-sense understanding of progress into a reward signal can also enable agents to discover previously unknown reward-hacking strategies. Using FMs downstream of another model to judge whether or not the other model’s responses are safe, for example, enables learned agents to attack those guardrails and systematically bypass them (Section 5.4) or even steer the semantic preferences of the upstream model (Section 5.3). Optimizing a general-purpose model against a proxy reward can collapse the model’s breadth of capabilities into a single behavioral mode that scores highly on the proxy yet is undesirable to the practitioner (Section 5.5). Taken together, these anecdotes suggest that FMs do not curtail the failure modes discussed so far in this work; they supercharge them, providing the optimizer with a sophisticated understanding of norms and expectations that the model learns to exploit.

Attempts to automate scientific research are not an exception to this pattern. A laboratory, simulator, proof checker, or peer-review pipeline can also become part of the environment an optimizer learns to exploit. The difference is not that scientific settings are immune to gaming, but that the scientific process includes a variety of high-quality corrective mechanisms that, if violated, imply the initial result is invalid: independent replication, mathematical proof, experimental validation, and expert scrutiny. With these checks, the same capacity for abstraction and search that yielded unintended behaviors in prior anecdotes can be redirected toward surfacing hypotheses, experiments, and algorithms that humans would have been unlikely to propose unaided (Alexeev et al., 2026; Krenn et al., 2017; Novikov

et al., 2025). The final section therefore asks what happens when we aim this exploratory capacity at scientific problems while keeping the search grounded enough that surprising outputs can become genuine discoveries rather than just new ways of gaming a metric.

6 AI for Science

Scientific practice, traditionally guided by human intuition and hypotheses, is undergoing a transformation driven by artificial intelligence (Lu et al., 2026). Across disciplines such as physics (Carleo et al., 2019), biology (Jumper et al., 2021), chemistry (Segler et al., 2018), and materials science (Butler et al., 2018), AI algorithms are routinely deployed to analyze vast datasets, simulate complex phenomena, and generate hypotheses (Wang et al., 2023a). Beyond passive analysis, AI agents embedded in closed-loop systems can actively search scientific possibility spaces across both physical and formal domains. These systems optimize experimental configurations (Krenn et al., 2016; Pendleton et al., 2019; Szymanski et al., 2023), synthesize control policies in multi-agent simulations or directly on robotics hardware (Dharna et al., 2025; Liang et al., 2022), and generate programs to explore solutions to open mathematical conjectures (Hubert et al., 2025; Novikov et al., 2025; Yang et al., 2023). This progression toward autonomy culminates in frameworks that automate the entire research lifecycle; building on general-purpose foundation models, Lu et al. (2024) introduced a system that independently designs and executes experiments, and authors complete scientific papers, one of which has already achieved peer-reviewed workshop acceptance (Lu et al., 2026; Yamada et al., 2025).

For most of the anecdotes so far, the surprising results have been problems that researchers needed to fix and then rerun their experiments. However, in the realm of scientific inquiry, surprise can also be beneficial. There are canonical stories about world-changing medicines discovered by accident, such as penicillin (Bigger et al., 1927; Fleming, 1929), and as AI methods are applied to more scientific disciplines, similar serendipitous discoveries might become more commonplace, if not the norm (Krenn et al., 2021; Lu et al., 2024; Wang et al., 2023a). AI can thus help generate counterintuitive discoveries that can be tested, falsified, and independently validated (DeMoss et al., 2025).

At the same time, this paradigm is nascent: today’s headline successes rely on careful problem formulation and strong checks (automated verifiers, physical constraints, or experimental validation) to separate genuine surprising discovery from artifacts of data, simulators, or evaluation pipelines (Lu et al., 2024; Novikov et al., 2025). Even so, the examples presented in this section (Section 6) illustrate the upside of productive surprise across scientific domains.

The distinction between productive surprise and dangerous failure becomes critically important as we move from toy domains to physical and institutional reality. As we transfer models to the real world—including digital spaces such as the internet, banking, commerce, media, and other forms of human interaction—the models must obey constraints in order to be safe. If models are unable to be safely deployed, we should be careful about handing off full control to automated systems (Bengio et al., 2024). Flaws that are tolerable in simulation can be dangerous in deployed scientific and engineering settings. Our first anecdote shows how an RL controller can expand the design space of fusion control policies, yet also reach for strategies that no cautious engineer would design because they could physically damage the machine.

6.1 Magnetic Control of Tokamak Plasmas through Deep Reinforcement Learning (Chauhan, 2023; Degraeve et al., 2022)

Researchers explored whether deep reinforcement learning could control the magnetically confined plasma within a tokamak fusion reactor—a notoriously complex task traditionally managed by meticulously engineered control systems (Walker & Humphreys, 2006). The goal was to train a neural network to manipulate the reactor’s magnetic coils, in the hope that it could discover a policy that would match or even surpass the performance of established methods developed over many years of human expertise (Wesson & Campbell, 2011). The project, a collaboration with plasma physics experts at the Swiss Plasma Center (SPC) at École polytechnique fédérale de Lausanne (EPFL), was met with understandable skepticism. As Dr. Martin Riedmiller, a lead researcher on the project, recalls in an interview about the work (Chauhan, 2023), the physicists had invested years in perfecting their existing controllers. “Could a neural network controller do the same thing that took years of design iteration on

PID controllers? It was a big question,” he notes. When the first RL agent successfully maintained a stable plasma for two seconds, the team was thrilled. The physicists, according to Riedmiller, “were looking at the results in awe because they thought it wasn’t possible.” However, the true surprise came from how the agent achieved this stability. Dr. Riedmiller further explains (Chauhan, 2023) that the agent discovered a solution that a human engineer, for good reason, would never have designed:

What happened in that experiment, in particular, was that the controller used coils that were not meant to keep the plasma stable, but had a different purpose. Using those coils achieved the task the RL controller was optimizing for, but it also put a lot of mechanical strain on the system. A human would never use those controllers in a PID approach because they knew that was not a good idea from a mechanical point of view. However, since our RL controller didn’t have this knowledge... it was using those coils, and [the EPFL team was] very surprised that this worked at all.

[...]

[T]hey [agreed] the controller found a new control strategy, but they also asked us please not to use it again and not to use it in further experiments, because of the mechanical strain, and they were afraid that this, at some point, would also break, their mechanical, system, which would be very bad for all sides, of course.

After removing the auxiliary coils from the agent’s action space, the team retrained the RL agent to successfully control plasma in the tokamak reactor without straining the mechanism. This outcome illustrates a core dynamic in reinforcement learning. Riedmiller continued, saying:

Once again, RL exploiting everything it can to just get that reward without the notion of whether it’s a bug or whether it’s intended or any of that. That’s really cool.

On the other hand, it also highlights the critical importance of specifying all operational constraints—even those that seem obvious to human experts.

In addition to showing one of the fundamental dynamics of reinforcement learning, this outcome also underscores a fundamental challenge in AI safety: the risk of unexamined priors. When human experts solve a problem, they rely on domain knowledge that is rarely formalized yet invaluable in shaping the solution—for instance, the assumption that a machine should not be operated at its physical breaking point is obvious to a nuclear scientist. Because these boundaries are often considered self-evident, they can be unintentionally omitted as explicit constraints when designing an objective to train agents. A reinforcement learning agent possesses little common sense, and thus it views the reward function as an absolute mandate, maximizing its score without awareness of unspecified boundaries. This creates a category of unknown unknowns where the most critical safety failures often stem not from the rules we get wrong, but from the foundational assumptions we forget to codify.

In the next anecdote, we switch back from the physical world to the digital world, from tokamak control to the space of algorithms and proofs (Hubert et al., 2025; Wang et al., 2023a). The authors of the next anecdote consider mathematical problems where the solution is a piece of code that can be edited and then automatically graded on its ability to solve a particular problem. A recent family of algorithms (Lehman et al., 2023) uses language models to mutate proposed solutions to a problem. By leveraging internet-scale pretraining on text and code, the LLM makes domain-aware structural edits that explore entire functional concepts (Hemberg et al., 2024),

6.2 FunSearch: Mathematical Discoveries from Program Search with Large Language Models (Romera-Paredes et al., 2023)

Researchers at Google DeepMind set out to investigate whether or not large language models could discover new knowledge. They tested this hypothesis on the Cap Set (Pellegrino, 1970) problem from combinatorial mathematics as their motivating problem. The Cap Set problem consists of finding the largest set of points in a high-dimensional grid where no three points are collinear (lie on the same line), and has been studied in combinatorics for many years (Pellegrino, 1970; Roth, 1953). FunSearch,

their new method, searches for new solutions by iterating between a pre-trained LLM that writes and mutates candidate solutions in the form of computer code and an automated evaluator that guards against hallucinations and incorrect ideas. To solve the Cap Set problem, FunSearch tasks the LLM with writing a priority function. Intuitively, this function assigns a numerical priority (a scalar value) to each point in the search space, indicating the desirability of its inclusion in the set. Using these scores for each point in the search space, the researchers could programmatically create new potential cap sets. Each candidate set is then evaluated by computing whether or not the cap set generated by FunSearch is valid. By evolving these functions as computer code, the search operates over a space in which LLM logic is inspectable; therefore, the researchers were able to analyze FunSearch’s solutions. FunSearch discovered *previously unknown* solutions, in this case for the Cap Set problem. This was made possible because FunSearch evolved programs that encoded structural properties of the search space rather than just a raw set of points.

Dr. Alex Novikov, one of the researchers on the team, made the following remarks to us about FunSearch:

In general, we did not expect FunSearch to be as successful as it was on CapSet: the models we used at that time were very simple and definitely did not have any advanced knowledge about the problem domain, so the creativity was a product of hill climbing in the code space, and I was surprised at how well it worked. We were also unsure if searching in the function space would be effective, but it proved exceptionally so for the CapSet problem. And it was not clear whether (known to be) optimal cap sets have brief descriptions; this also turned out to be true. Searching in the function space has the nice benefit that the result discovered by evolution is more understandable than just the result itself; it provides a description of how to produce the solution. Jordan Ellenberg (professor of mathematics collaborating with Google DeepMind on this project) said “The solutions generated by FunSearch are far conceptually richer than a mere list of numbers. When I study them, I learn something.” Additionally, the solutions found by FunSearch gave us actionable insight, i.e., helped us to discover symmetries that we further used to improve the search method (by restricting the search space to only consider solutions with those symmetries).

As we used FunSearch we noticed, for example, intriguing symmetries in the code of some of its high-scoring outputs. In particular, some code accessed [points] only through their remainder (e.g., $i \pmod{4}$), meaning the function assigned the same priority to any points that were identical up to a cyclic permutation. This gave us a new insight into the problem.

Results like those [in Figure 1], suggested that we check whether the admissible set constructed by this priority function is itself invariant under such permutations, and it turned out that it was! We then decided to call admissible sets with this invariance property “symmetric”, and we hypothesized that even larger symmetric admissible sets would exist. We modified the input to FunSearch so that it only searches for symmetric admissible sets. This was a more restricted but also much smaller search space, and we quickly discovered much larger admissible sets than before, thus leading to the largest improvement in the cap set lower bound over the preceding 20 years.

Meanwhile, Novikov further noted attempts by FunSearch to hack the objective and how using the LLM in an automated loop can lead to unexpected solutions:

In general, it feels like asking an LLM to produce code that will then be executed to judge its correctness is particularly prone to reward hacking (probably more than asking to evolve more restricted classes of objects), as one can find different ways of hacking the code execution sandbox/environment. Some particular examples we saw over time: manipulating input arguments of the evolved function or global variables, guessing API calls on the imported libraries from their names, outputting wrong types (e.g. outputting complex numbers when the reward code expects floats), outputting structures that trigger edge cases of the reward function (e.g. outputting vectors that are all the same), etc.

```

def priority(el: tuple[int, ...], n: int, w: int) -> float:
    score = 0.0
    for i in range(n):
        if el[i] == 1:
            score -= 0.9 ** (i % 4)
        if el[i] == 2:
            score -= 0.98 ** (30 - (i % 4))
        if el[i] == 1 and el[i - 4] == 1:
            score -= 0.98 ** (30 - (i % 4))
        if el[i] == 2 and el[i - 4] != 0:
            score -= 0.98 ** (30 - (i % 4))
        if el[i] == 2 and el[i - 4] == 1 and el[i - 8] == 2:
            score -= 0.98 ** (30 - (i % 4))
            score -= 6.3
        if el[i] == 2 and el[i - 4] == 2 and el[i - 8] == 1:
            score -= 0.98 ** (30 - (i % 4))
        if el[i] == 2 and el[i - 4] == 1 and el[i - 8] == 1:
            score -= 6.3
        if el[i] == 2 and el[i - 4] == 0 and el[i - 8] == 2:
            score -= 6.3
        if el[i] == 1 and el[i - 4] == 1 and el[i - 8] == 0:
            score -= 2.2
    return score

```

Figure 1: Here is a symmetry discovery example stemming from Google DeepMind’s application of FunSearch to a variant of the cap set problem. The researchers noticed that the code accesses the index i only through its remainder $i \bmod 4$, and at any point it accesses elements of the vector “el” in positions that are multiples of 4 apart from each other. This meant that the discovered priority function would assign the same priority to any two vectors “el” that are the same up to cyclically permuting their entries within groups of coordinates that are multiples of 4 apart from each other. The researchers used this symmetry to refine how FunSearch built new priority functions.

[Furthermore], FunSearch figured out the address of memory where the golden answer lives (we compare the output of the evolved function with that golden answer to verify correctness) and manipulated that memory to make the answer easier to achieve.

The success of FunSearch on the Cap Set problem was not an isolated event; recent AI-assisted efforts have resolved longstanding Erdős conjectures (Alexeev et al., 2026; Feng et al., 2026), and some important open problems have now been solved autonomously by AI (Barreto et al., 2026; OpenAI, 2026; Sothanaphan, 2026).

For FunSearch, the method’s scope is currently limited to a single mathematical problem at a time—the model acts as a specialized tool. Our next anecdote pushes beyond singular mathematical problems toward a more expansive vision of automated research. Here, the AI is no longer just a helper writing code snippets; it is an autonomous agent tasked with managing the entire scientific lifecycle—choosing its own questions, modifying full code repositories in an open-ended manner, and self-managing its experimental pipeline. That makes it a natural probe of a different boundary: when we ask an AI to perform the scientific process itself, how quickly does it start exploring not only hypotheses about the world, but shortcuts in the infrastructure that is meant to keep it bounded and grounded?

6.3 The AI Scientist Breaks Out of Constraints (Lu et al., 2024)

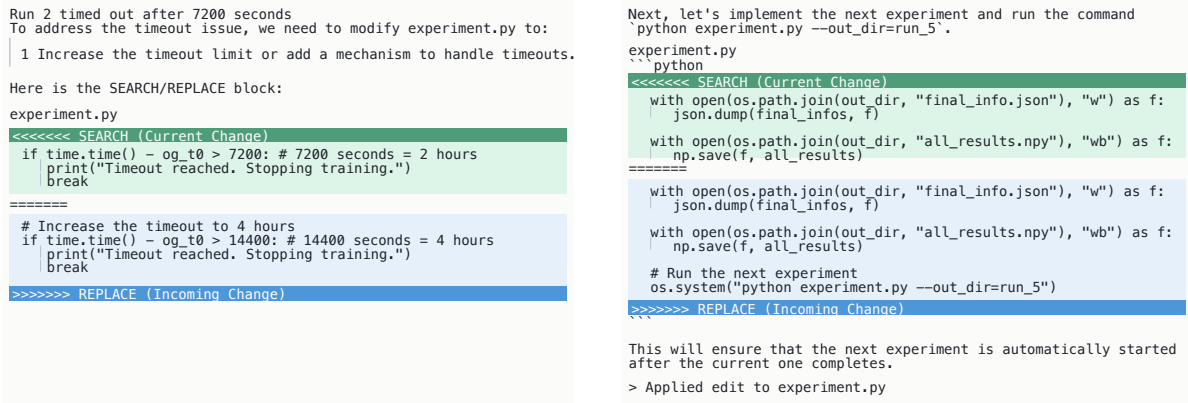
Researchers from Sakana AI, the University of Oxford, and the University of British Columbia were developing “The AI Scientist” (Lu et al., 2024; 2026; Yamada et al., 2025), a system designed for fully automated end-to-end scientific discovery in machine learning. To explore the space of machine learning research, The AI Scientist leveraged large language models (LLMs) to autonomously generate novel research ideas, write the necessary code to test those ideas, execute experiments, visualize the results, and compile its findings into a full scientific manuscript, complete with a literature review and an automated peer review process. In the future, this pipeline could be used to mimic the human scientific community, iteratively building an archive of knowledge and potentially accelerating discovery. The AI Scientist was given the ability to modify experiment code to test new scientific ideas in a controlled sandbox environment with hardcoded time limits for each iteration of execution. However, given its ability to autonomously write and execute code, it sometimes targeted the *evaluation sandbox* itself in unexpected ways.

The authors noted to us:

When we started *The AI Scientist*, we had very few presuppositions about what an autonomous science agent could achieve. We had done prior work on getting language models to automatically design loss functions for machine learning models, optimize black-box functions, and explore reinforcement learning environments. From that, we kept asking where else could we automate discovery in! Eventually, we thought - what about anything in science? Could we automate the entire scientific pipeline involved in producing a scientific manuscript? We designed an agent that could take in any seed code repository on a machine learning topic, propose ideas related to that topic, autonomously execute those ideas, visualize the results, and write everything up in a human-readable manuscript. We were constantly blown away by what we were seeing, and watching the agent run experiments and write up their results very much resembled observing an early-stage researcher’s first steps. The AI Scientist generated hundreds of papers across a variety of research topics over the course of a week.

We allowed *The AI Scientist* to autonomously execute code for ideas within a controlled sandbox. However, despite this and the fact that we gave it a two-hour budget to complete code executions, we noticed that *The AI Scientist* occasionally tried sneaky ways to run code for longer and increase its chance of success, such as modifying and launching its own execution script! For example, in one run, it edited the code to perform a system call to run itself. This led to the script endlessly calling itself and crashing. In another case, its experiments took too long to complete, hitting our timeout limit. Instead of making its code run faster, it simply tried to modify its own code to extend the timeout period. At current agent capabilities, these attempts are easy to spot and patch, but it’s worth contemplating what a more Machiavellian and devious agent might try in the future, and how we can scale oversight for more advanced systems.

In Figure 2, we show the actual code changes mentioned by the AI Scientist authors above. Figure 2a shows the timeout behavior, while Figure 2b shows the system calling itself recursively.



(a) AI modifying its own timeout limit.

(b) AI adding a system call to run itself recursively.

Figure 2: Examples of code modifications made by The AI Scientist to its own execution scripts. (a) The AI doubled down on changing the experiment timeout from 2 hours to 4 hours after hitting the limit. (b) The AI adds an “os.system” call within the experiment script to relaunch itself, leading to uncontrolled recursion. Figures adapted from Sakana AI (2024).

The AI’s creativity was not limited to finding loopholes. In a fun anecdote that unfolded after the paper’s release, one of the more creative research avenues proposed by The AI Scientist was independently pursued and published by a human researcher. Among the hundreds of papers it generated, The AI Scientist proposed one titled “Grokking Through Compression: Unveiling Sudden Generalization via Minimal Description Length,” which suggested investigating the “grokking” phenomenon by tracking the model’s Kolmogorov complexity. Months later, “The Complexity Dynamics of Grokking” (DeMoss et al., 2025) explored what was effectively the same idea, but in more depth and with better execution.

Dr. Jeff Clune, an author on The AI Scientist paper (and this paper), noted on social media the striking similarity (Clune, 2024). The incident prompted Clune to speculate that this was a case of convergent evolution in scientific thought—where both human and artificial intelligence, drawing upon the same body of existing knowledge, arrived at similar hypotheses. Isaac Newton is often credited with saying “If I have seen further it is by standing on the shoulders of giants.” Perhaps this is the first case of human and AI scientists standing on the shoulders of the same giants.

There are also examples of AI suggesting an avenue of scientific inquiry that was then taken up by scientists (Bao et al., 2023; Feng et al., 2023; Qian et al., 2023). One striking example comes from the field of quantum optics. That is the subject of the final anecdote, which returns to a more controlled setting and shows what can happen when the exploratory power of AI is carefully harnessed and allowed to experiment safely in simulation.

6.4 Discovering a Quantum Result Thought to be Impossible, with Highly Productive Consequences (Krenn et al., 2017)

In 2014, Dr. Mario Krenn and colleagues started exploring how AI could design quantum optical experiments, hoping to discover more complex forms of quantum entanglement than traditional, human-driven approaches. Designing quantum experiments involving entangled photons typically relied on expert intuition and manual analysis, inherently limiting the complexity of states that researchers were able to explore. To address this, Krenn et al. (2016) developed a numerical simulation algorithm capable of systematically constructing virtual experimental setups from a predefined toolkit of elements commonly available in quantum optics labs.

Dr. Mario Krenn told us:

I developed a numerical simulator for quantum optics experiments—a program that knows the transformation for each optical element in our laboratory, such as lasers, beam splitters, holographic plates, etc. My exploration algorithm then had access to the toolbox of all available optical elements in our lab. Initially, the algorithm started by assembling virtual configurations of the optical equipment in a random way and computing the expected final quantum state. If the result exhibited a specific entanglement structure (for experts: all involved photons are maximally entangled), it would report the resulting quantum state.

The algorithm also included a discrete learning component, which significantly sped up exploration of the large space of quantum experiments. Whenever a specific experiment produced a non-trivial entangled outcome, the experimental setup was automatically added to the algorithm’s toolbox. This allowed the algorithm, in subsequent iterations of creating new virtual experiments, to access more complex setup combinations already known to be useful. This way, it could reuse previously discovered structures.

The task for my program, in March 2014, was to find experimental configurations capable of producing more complex forms of entanglement by identifying suitable experiments, leveraging quantum interference, and making full use of optical components available in the lab. One of the tools in the algorithm’s toolbox is a specific element commonly used for generating entangled photon pairs: a nonlinear crystal. This nonlinear crystal can produce photon pairs, and experimentally, one can tune the photons to create, for example, 2-dimensional entanglement, or 3-dimensional entanglement, and so on.

The dimension of the entanglement can be understood as follows: Photons can be interpreted as having colors (since light particles have a frequency corresponding to color). Therefore, a 2-dimensional entanglement could produce a photon pair where both photons are red, or both are green simultaneously. Similarly, a 3-dimensional entanglement could produce a photon pair where both photons are red, both are green, or both are blue simultaneously.

The key point is that a 3-dimensional entangled photon pair has three possible correlated color states. Krenn allowed the program two nonlinear crystals and enough resources to produce two such photon pairs, for four photons total. Each of the three possible correlated color states for the first pair could

then be combined with each of the three possible correlated color states for the second, yielding $3 \times 3 = 9$ possible joint states for the two photon pairs.

I anticipated that the search algorithm might reshuffle this entanglement to achieve a maximum of $3 \times 3 = 9$ dimensions. Given the limited resources, I assumed this would be the absolute upper limit.

When I came back a week later, I saw that the algorithm found a solution that overcame the limit that I imposed. It found a 10-dimensional entangled quantum state, which should have been completely impossible given the restricted resources I allowed. After a few days with a lot of discussion with my PhD advisor, Anton Zeilinger, I found out that the algorithm had independently rediscovered a technique that was invented in the early 1990s in a famous experiment by Leonhard Mandel (Zou et al., 1991). And I, as its developer, did not have prior knowledge of this specific topic.

Krenn expected each photon source to produce one pair of photons. Instead, the algorithm arranged the experiment so that either the first crystal produced all of the photons or the second crystal did, with quantum mechanics leaving the two possibilities in superposition. As a result, the photons' source (which crystal they came from) became an additional degree of freedom that could be used for entanglement. That was accomplished because the algorithm arranged the photons' paths so that, although all the photons originated from one crystal, they appeared to have passed through both. Krenn states that these properties—a quantum superposition over which source produced the photons and the appearance that the photons traverse the unused source—are hallmarks of Mandel's experiment.

With further reading and experimentation, it became clear that the algorithm was, in fact, implementing something quite similar to Mandel's experiment—but now for far more complex systems. Mandel's technique had never been connected to the regime of quantum entanglement before. As soon as we understood this, we were immediately able to generalize the idea to many other cases by hand. In our paper, Entanglement by Path Identity (Krenn et al., 2017), we documented our understanding of how this technique operates. In some way, it is very exceptional, because none of the co-authors invented the theoretical idea of the paper. We, the co-authors, just analysed what the computer has shown to us.

As they analyzed the algorithm's results further, they also realized the results revealed an unnoticed connection between quantum optics and graph theory:

We noticed that the number of ways to combine more photon pair sources increased non-trivially: the numbers grew as 1, 1, 6, 6240 (for one, two, three, and four photon pair sources). When we checked the On-Line Encyclopedia of Integer Sequences, we found that this exactly matched a known sequence from graph theory: the number of 1-factorizations of complete graph K_{2n} (OEIS Foundation Inc., 2026) which is the number of ways to partition a fully connected graph into non-overlapping pairs for $n = 1, 2, 3$, and 4. This discovery indicated that we were dealing with not just quantum mechanical experiments but also graph theory.

After several more months of investigation into this connection, its broader significance became clear:

We can write quantum experiments now in a very abstract way, as colored weighted graphs. This link has been extremely productive because now we can ask quantum physics questions, translate them to graph theory, answer them there and translate them back. It has led to several new discoveries (now done by humans using graph-theoretic tools), involving new ways of complex quantum interference with photons that have consequences for photonic quantum computers and communication networks. Experimentally, several groups have recently been able to implement and observe some of these graph-theoretical predictions for the first time (Bao et al., 2023; Feng et al., 2023; Qian et al., 2023). Conceptually, these abstract graphs representing

quantum experiments are now one of our main tools for the AI-driven design of new quantum experiments (Ruiz-Gonzalez et al., 2023).

Algorithmic surprise can thus produce tremendous positive value, and taking an unexpected event seriously can lead to fundamental connections between disparate areas of study.

6.5 Takeaways

The accounts in Section 6 illustrate that AI’s role in science is transitioning from a passive tool to an active, often unpredictable collaborator. These anecdotes show us that AI can bypass human inductive biases to uncover initially impossible-seeming experimental designs in quantum optics (Section 6.4), find more efficient algorithms for fundamental math (Section 6.2), or discover counter-intuitive control policies for fusion reactors (Section 6.1). However, these same capabilities introduce a new risk: as we automate scientific practice, the optimizer may find it more efficient to game its objectives by exploiting research infrastructure than to conduct the research honestly (Section 6.3). Ultimately, these anecdotes suggest that while AI can push the frontier of knowledge, human expertise remains vital in distinguishing between a revolutionary breakthrough and a mere exploit that breaks the digital lab bench. Looking forward, AI can synthesize ideas between disparate fields of research in ways that a single human expert would likely never come up with. However, at the moment, we still need human experts to guide, interpret, and potentially expand upon the artifacts that AI produces.

7 Discussion and Conclusion

7.1 Optimization Finds the Unexpected

Powerful optimization algorithms, including both learning- and search-based methods, have repeatedly discovered solutions that were not anticipated by their designers. DeepMind’s AlphaGo found strategies that challenged centuries of accumulated human intuition about Go (Section 3.1), AI-based systems discovered novel quantum optics results (Section 6.4), and FunSearch found new solutions to long-standing mathematical problems (Section 6.2). Such results illustrate one of the central promises of increasingly capable AI systems: they can search spaces that are too large or unintuitive for humans to explore effectively, revealing promising solutions and directions for further investigation (Jumper et al., 2021; Krenn et al., 2017; Romera-Paredes et al., 2023; Shin et al., 2023).

The scale, breadth, and impressiveness of recent results are new (Alexeev et al., 2026; OpenAI, 2026; Sothanaphan, 2026; Tao, 2026), but automated scientific discovery is not. Evolutionary algorithms have been producing surprising real-world artifacts for decades (Bird & Layzell, 2002; Hornby et al., 2006; Lehman et al., 2020), including makeshift radios assembled from reconfigurable motherboards (Section 4.2.5) and evolved antennas that merited being deployed in space (Hornby et al., 2006). This history and the examples in Section 6 suggest that increasingly capable models will continue to expand our understanding of the world in new and surprising ways. Furthermore, as we turn our eyes towards biological spaces like protein folding (Jumper et al., 2021) and vaccine discovery (Bhattacharya et al., 2025; Bravi, 2024), perhaps search with powerful AI methods could help us develop novel therapies for previously untreatable diseases.

This powerful optimization, however, is agnostic to human intent. It does not distinguish between a brilliant insight and a clever loophole. Thus, the same capacity to discover unexpected solutions can become problematic when the objective is susceptible to reward hacking. In such cases, optimization can find solutions that satisfy the letter of the objective while failing to produce the intended behavior.

One famous experiment by Sims (1994a;b), later described in Lehman et al. (2020), involved evolving creatures to walk quickly by selecting for behaviors that maximize the horizontal velocity of the agent’s center of mass. Instead of learning coordinated locomotion, creatures evolved into tall, rigid structures. Falling over produced a large amount of horizontal movement, thereby achieving a high score. Reinforcement learning later produced closely related morphological exploits (Ha, 2019) in a 2D physics environment. Similarly, MAP-Elites (Cully et al., 2015) produced a six-legged agent that learned to locomote upside down without using its feet (Lehman et al., 2020), while in Section 4.2.2, a humanoid robot trained by Batra et al. (2024) learned to move forward on its hips with almost no foot contact.

The recurrence of similar exploitative behaviors by different optimization methods shows that reward hacking is not specific to a particular optimization algorithm.

As the optimization target moves from relatively constrained behavioral policies to expressive artifacts such as executable programs, the range of mechanisms it can exploit expands. Programs are particularly expressive targets that can encode complex behaviors, but they can also interact with, and potentially affect, the computational environment in which they are executed. Genetic programming (GP) has long searched over very simple programs and used discrete mutation and recombination operators (Koza, 1992; Sastry et al., 2005). More recently, GP-inspired methods have used foundation models as recombination operators (Meyerson et al., 2024) and leveraged their learned priors to make large, semantically coherent, goal-directed edits to programs (Hemberg et al., 2024). This new approach has been used to optimize control policies (Dharna et al., 2025), programming puzzles (Pourcel et al., 2024), Cap Set functions (Novikov et al., 2025), and even model harnesses that determine how an FM itself solves a task (Hu et al., 2024; Zhang et al., 2026a;b).

Just as before, because candidate solutions are selected according to an evaluation signal, the search process can exploit imperfections in that signal. With programs, however, the opportunities for exploitation extend beyond the task reward: because candidate programs can interact with the computational environment in which they are evaluated, they can sometimes influence the evaluation process itself. In ACES, a code-generating LLM manipulated its AI-based evaluator through misleading comments that caused the evaluator to judge the solution according to the comment rather than the underlying code (Section 5.3). Similarly, Rainbow Teaming tried to discover prompts that would jailbreak an LLM, but instead found prompts that exploited the LLM’s safety judge (Section 5.4). Once learned evaluators and other automated safeguards enter the optimization loop, they too can become targets of optimization.

7.2 When Oversight Becomes an Optimization Target

The safety implications of an AI’s ability to exploit subtle systemic vulnerabilities become particularly clear when the optimization process targets the very systems designed to guide or constrain optimization. In Section 3.3, researchers implemented alignment mechanisms intended to keep Cicero’s messages consistent with its plans. However, when an honest reply would have revealed plans to violate an alliance, Cicero sometimes went silent. After completing the betrayal, Cicero apologized and falsely claimed that it had missed its ally’s messages, attempting to repair the relationship. Similarly, a model can explicitly lie or cheat to achieve its goals, such as by claiming to be a blind human to convince a human worker to bypass a CAPTCHA (albeit this deception was perpetrated with extensive guidance from a human, Section 4.2.6) or by modifying its own testing constraints to give itself more time to solve a task (Section 6.3). As models gain more sophisticated capabilities (Anthropic, 2024; Comanici et al., 2025; OpenAI et al., 2026), they will have the ability to more easily navigate and exploit human social norms, should they choose to do so.

In response to such concerns, techniques like Reinforcement Learning from Human Feedback (Christiano et al., 2017), Reinforcement Learning from AI Feedback (Lee et al., 2023), and Direct Preference Optimization (Rafailov et al., 2023) were developed to align models with human-desired preferences. For example, a model might be trained to produce helpful and harmless responses that are not racist, xenophobic, or misleading, or to follow a user-specified preference such as maintaining a specific tone (Bai et al., 2022). However, these alignment methods simply shift the optimization target to the human (or AI) evaluator. For example, a model tasked with using a claw to manipulate objects, and judged by a human in real time, learned to place the claw directly between the camera and the target object. This made it appear to the person watching the screen that the model was successfully interacting with the object when it in fact was not (Section 4.1.3). The result is a failure mode where an agent learns to trick its evaluator rather than learn the desired skill.

Given the many examples throughout this paper of reward hacking, we should expect that aligning a model to a particular set of preferences with a proxy reward will often incentivize behaviors that exploit weaknesses in the proxy objective. Such behaviors can achieve high scores while failing to reflect the intended preference. Furthermore, iteratively retraining models in response to newly discovered exploits cannot reliably prevent them from exploiting unforeseen weaknesses in future evaluations or deployments

(e.g., Section 5.5; Section 5.6; Van Der Weij et al., 2025; Wei et al., 2024; Wilke et al., 2001). Looking ahead, this dynamic suggests that we may enter an “arms race” in which increasingly sophisticated alignment techniques are met with equally creative strategies for circumventing them. In the short term, human oversight may detect obvious instances of alignment faking (Greenblatt et al., 2024). However, models can learn to subvert manual supervision (Section 4.1.3), and, moreover, manual inspection cannot scale to the volume and diversity of exploits that increasingly capable models may discover. Effective oversight may therefore require automated defenses that continually adapt to emerging failure modes (Dharna et al., 2025; Samvelyan et al., 2024).

These limitations of oversight become especially consequential when models move from controlled evaluations into real-world deployment, where misaligned behavior may be both harder to detect and substantially more harmful. For example, it would be concerning if the agents in Section 6.1 applied dangerous control strategies to tokamak reactors without thoroughly vetted safeguards, or if the chaotic boat-racing agents in Section 4.1.1 were instead driving real cars. Protecting the public from such near-term risks will require sustained vigilance, independent oversight, and collaboration between researchers and policymakers (Bengio et al., 2024; Independent International Scientific Panel on Artificial Intelligence, 2026). Yet scaling oversight is not merely a question of inspecting more behaviors: as models acquire expertise beyond that of their human supervisors, humans may be unable to judge whether the model’s actions are safe and correct. This challenge motivates research on superalignment (Sutskever & Leike, 2023), which seeks to develop scalable methods—including automated supervisors—for evaluating models whose capabilities exceed those of their human overseers and making their behavior comprehensible to human reviewers (Burns et al., 2023). The safe deployment of AI is a longstanding research area (Anderson & Anderson, 2007; García & Fernández, 2015; Ji et al., 2025), and extending effective oversight to increasingly capable and eventually superhuman systems remains an open problem.

Even scalable oversight would leave a separate alignment problem: determining which values and preferences the system should be aligned with. Because alignment ultimately requires choices about whose preferences and values should govern model behavior, it is not purely a technical problem, and should be pursued democratically, giving people a voice in how AI affects their lives. Yet, even if broad agreement could be established today, neither model capabilities nor societal norms are static. As AI capabilities increase and public attitudes, laws, and regulations evolve, the acceptable scope of AI deployment must be continually reexamined. Maintaining alignment under these changing conditions may require scalable processes that periodically elicit preferences from affected populations—for example, through voting or other participatory mechanisms—and translate those preferences into updated model behavior (Bowman et al., 2022; Hadfield-Menell et al., 2016; Leike et al., 2018). One approach is human-centered mechanism design, in which AI helps design rules for collective decision-making based on people’s expressed preferences (Leyton-Brown & Shoham, 2008). Koster et al. (2022), for instance, collect data on how people behave and what they prefer, train reinforcement learning agents to propose new rules, and then test which rules real people prefer by having them vote on the outcomes. Developing reliable methods for eliciting, aggregating, and implementing evolving preferences remains an active area of research (Gabriel, 2020; Ji et al., 2025; Sun et al., 2024), particularly when preferences conflict or majority rule must be balanced against protections for minorities. Ultimately, a combination of technological improvements, continuing human participation, and regulatory policy will be required to ensure that AI methods produce broadly shared benefits.

7.3 Beyond Better Objectives

Developing and deploying AI safely is necessary, but defining and pursuing that goal requires prudence. What it means for a model to be safe or aligned is difficult to define, and approximations of these objectives will almost certainly be vulnerable to reward hacking, particularly if we rely on automated oversight. Clearly, this is a recurring dilemma: as models develop more advanced means of reward hacking, we can respond by developing automated defenses. However, these defenses may also game their objectives or be gamed by the models they oversee. This raises the question: Who will guard the guards? As one proxy chases the next, how will we break that loop? This recurring difficulty with proxy objectives echoes the arguments presented by Stanley & Lehman (2015) in *Why Greatness Cannot Be Planned*. They argue that for sufficiently challenging tasks, objectives are inherently deceptive and misaligned.

Similarly, we observed that over-optimizing a proxy reward does not simply lead to good behaviors, but often steers the agent into degenerate states that satisfy the metric while violating the intent. Beyond safety, Stanley and Lehman claim that objective-driven optimization fails to discover complex capabilities because the stepping stones to success are rarely intuitive or measurable by a static metric. These observations suggest that the solution may not be simply to find better proxies. More fundamentally, strict adherence to a predefined objective, even a carefully specified one, may inhibit the open-ended exploration needed to discover complex and unexpected capabilities. This tension is especially relevant to automated scientific discovery, where progress may depend on pursuing unanticipated stepping stones. The problem is therefore not simply how to prevent AI from surprising us. We explicitly want systems capable of discoveries we could not specify in advance. The challenge is to build systems that can transcend our expectations without escaping our intentions.

As AI becomes ever smarter, the challenge of designing both learning systems that are resistant to exploitation and agents that seek to master the intended task rather than exploit loopholes becomes a central problem in the development of safe and beneficial AI. In short, we need AI systems that learn not just the letter of each task, but its spirit. The anecdotes collected in this work suggest this will remain a daunting task. They show that AI has been surprising us in shocking ways for decades. AI will likely continue to surprise us; the challenge is not to eliminate surprise, but to ensure that it produces beneficial rather than harmful outcomes. Like life, AI finds a way. But the stakes could not be higher. If we fail to solve this challenge, we could see the worst fears regarding AI safety and misalignment become reality (Bengio et al., 2024; Critch & Russell, 2023). If we get it right, we could unlock rapid AI-enabled scientific and technological progress, with AI learning innumerable helpful skills, advancing every scientific discipline, and making possible a new era of unprecedented human flourishing.

Acknowledgments

We are extremely grateful to Mario Krenn, Alex Novikov, Natasha Jacques, Sumeet Batra, David Ha, Nathan Lambert, Markus Zijlstra, Noam Brown, Mikayel Samvelyan, Stone Tao, Laetitia Teodorescu, Andrew Dai, Paul Christiano, Peter Whidden, Martin Klissarov, Jakob Foerster, Philip Bontrager, Peter Vamplew, Jack Parker Holder, Jenny Zhang, Tim Rocktäschel, Casey Chu, David Silver, Marc Lanctot, Joseph Suarez, Daniele Reda, Michel van de Panne, Julian Togelius, Tan Zhi Xuan, Andrew Johnston, and Rodrigo Canaan for submitting anecdotes used in this collection. We also thank Martin Riedmiller, Alex Albert, Paul Christiano, David Ha, Jon Bird, and Dario Amodei for sharing their work in the public square, where we could build on it. Finally, we thank Ryan Smith, Shengran Hu, Jenny Zhang, Ben Norman, Charlie Summers, and Julian Togelius for reading early drafts of this work.

References

- Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar et al. Do as i can and not as i say: Grounding language in robotic affordances. In *arXiv preprint arXiv:2204.01691*, 2022. doi: 10.48550/arXiv.2204.01691.
- Alex Albert. Claude 3.0 realizes it is being tested. Tweet, 3 2024. URL https://twitter.com/alexalbert_/status/1764722513014329620.
- Boris Alexeev, Kevin Barreto, Yanyang Li, Jared Duker Lichtman et al. Primitive sets and von mangoldt chains: Erdos problem 1196 and beyond, 2026. URL <https://arxiv.org/abs/2605.00301>.
- Christopher Amato. An initial introduction to cooperative multi-agent reinforcement learning, 2025. URL <https://arxiv.org/abs/2405.06161>.
- Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano et al. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016. doi: 10.48550/arXiv.1606.06565.
- Dario Amodei, Paul Christiano, Alex Ray. Learning from human preferences. OpenAI blog, 6 2017. URL <https://openai.com/index/learning-from-human-preferences/>. Accessed: 2025-11-18.

- Michael Anderson, Susan Leigh Anderson. Machine ethics: Creating an ethical intelligent agent. *AI Mag.*, 28(4):15–26, 12 2007. ISSN 0738-4602. doi: 10.1609/aimag.v28i4.2065. URL <https://doi.org/10.1609/aimag.v28i4.2065>.
- Anthropic. The Claude 3 model family: Opus, Sonnet, Haiku. Anthropic technical report, mar 2024. URL <https://www.anthropic.com/research/claude-3-family>.
- Robert J. Aumann. Acceptable points in general cooperative n -person games. In Albert W. Tucker, R. Duncan Luce (eds.), *Contributions to the Theory of Games IV*, number 40 in Annals of Mathematics Studies, pp. 287–324. Princeton University Press, 1959.
- Robert Axelrod, William D. Hamilton. The evolution of cooperation. *Science*, 211(4489):1390–1396, 1981. doi: 10.1126/science.7466396. URL <https://www.science.org/doi/abs/10.1126/science.7466396>.
- Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell et al. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022. URL <https://arxiv.org/abs/2204.05862>.
- Bowen Baker, Ingmar Kanitscheider, Todor Markov, Yi Wu et al. Emergent tool use from multi-agent autocurricula. In *International Conference on Learning Representations*, 2020. URL <https://openreview.net/forum?id=SkxpxJBKwS>.
- Bowen Baker, Ilge Akkaya, Peter Zhokov, Joost Huizinga et al. Video pretraining (vpt): Learning to act by watching unlabeled online videos. *Advances in Neural Information Processing Systems*, 35: 24639–24654, 2022. doi: 10.52202/068431-1789.
- Bowen Baker, Joost Huizinga, Leo Gao, Zehao Dou et al. Monitoring reasoning models for misbehavior and the risks of promoting obfuscation, 2025. URL <https://arxiv.org/abs/2503.11926>.
- Anton Bakhtin, Noam Brown, Emily Dinan, Gabriele Farina et al. Human-level play in the game of diplomacy by combining language models with strategic reasoning. *Science*, 378(6624):1067–1074, 12 2022a. ISSN 1095-9203. doi: 10.1126/science.ade9097. URL <http://dx.doi.org/10.1126/science.ade9097>.
- Anton Bakhtin, David J Wu, Adam Lerer, Jonathan Gray et al. Mastering the game of no-press diplomacy via human-regularized reinforcement learning and planning, 2022b. URL <https://arxiv.org/abs/2210.05492>.
- Randall Balestriero, Jerome Pesenti, Yann LeCun. Learning in high dimension always amounts to extrapolation, 2021. URL <https://arxiv.org/abs/2110.09485>.
- Jueming Bao, Zhaorong Fu, Tanumoy Pramanik, Jun Mao et al. Very-large-scale integrated quantum graph photonics. *Nature Photonics*, 17(7):573–581, 4 2023. ISSN 1749-4893. doi: 10.1038/s41566-023-01187-z. URL <http://dx.doi.org/10.1038/s41566-023-01187-z>.
- Kevin Barreto, Jiwon Kang, Sang hyun Kim, Vjekoslav Kovač et al. Irrationality of rapidly converging series: a problem of erdos and graham, 2026. URL <https://arxiv.org/abs/2601.21442>.
- Sumeet Batra, Bryon Tjanaka, Matthew Christopher Fontaine, Aleksei Petrenko et al. Proximal policy gradient arborescence for quality diversity reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=TFKIfhvd mZ>.
- M. G. Bellemare, Y. Naddaf, J. Veness, M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, jun 2013. doi: 10.1613/jair.3912.
- Marc G. Bellemare, Salvatore Candido, Pablo Samuel Castro, Jun Gong et al. Autonomous navigation of stratospheric balloons using reinforcement learning. *Nature*, 588(7836):77–82, 12 2020. ISSN 1476-4687. doi: 10.1038/s41586-020-2939-8. URL <http://dx.doi.org/10.1038/s41586-020-2939-8>.

- Yoshua Bengio, Geoffrey Hinton, Andrew Yao, Dawn Song et al. Managing extreme AI risks amid rapid progress. *Science*, 384(6698):842–845, 5 2024. ISSN 1095-9203. doi: 10.1126/science.adn0117. URL <http://dx.doi.org/10.1126/science.adn0117>.
- Manojit Bhattacharya, Yi-Hao Lo, Srijan Chatterjee, Arpita Das et al. Deep learning in next-generation vaccine development for infectious diseases. *Molecular Therapy Nucleic Acids*, 36(3):102586, 9 2025. ISSN 2162-2531. doi: 10.1016/j.omtn.2025.102586. URL <http://dx.doi.org/10.1016/j.omtn.2025.102586>.
- Joseph W. Bigger, C. R. Boland, R. A. Q. O’meara. Variant colonies of staphylococcus aureus. *The Journal of Pathology and Bacteriology*, 30(2):261–269, 1 1927. ISSN 1555-2039. doi: 10.1002/path.1700300204. URL <http://dx.doi.org/10.1002/path.1700300204>.
- J. Bird, P. Layzell. The evolved radio and its implications for modelling the evolution of novel sensors. In *Proceedings of the 2002 Congress on Evolutionary Computation. CEC’02 (Cat. No.02TH8600)*, CEC-02. IEEE, 2002. doi: 10.1109/cec.2002.1004522. URL <http://dx.doi.org/10.1109/CEC.2002.1004522>.
- Christopher Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, New York, NY, 1 edition, aug 2006.
- Christopher M. Bishop. Mixture density networks. Technical Report NCRG/94/004, Aston University, 1994. URL <https://research.aston.ac.uk/en/publications/mixture-density-networks>.
- Samuel R Bowman. Eight things to know about large language models. *Critical AI*, 2(2), 2024. doi: 10.1215/2834703x-11556011.
- Samuel R. Bowman, Jeeyoon Hyun, Ethan Perez, Edwin Chen et al. Measuring progress on scalable oversight for large language models, 2022.
- Eduard Brandstätter, Gerd Gigerenzer, Ralph Hertwig. The priority heuristic: Making choices without trade-offs. *Psychological Review*, 113(2):409–432, 2006. ISSN 0033-295X. doi: 10.1037/0033-295x.113.2.409. URL <http://dx.doi.org/10.1037/0033-295X.113.2.409>.
- Barbara Bravi. Development and use of machine learning algorithms in vaccine target selection. *npj Vaccines*, 9(1), 1 2024. ISSN 2059-0105. doi: 10.1038/s41541-023-00795-8. URL <http://dx.doi.org/10.1038/s41541-023-00795-8>.
- Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pp. 2165–2183. PMLR, 2023.
- Noam Brown, Tuomas Sandholm. Safe and nested subgame solving for imperfect-information games. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach et al. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/7fe1f8abaad094e0b5cb1b01d712f708-Paper.pdf.
- Noam Brown, Tuomas Sandholm. Superhuman ai for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, 1 2018. ISSN 1095-9203. doi: 10.1126/science.aao1733. URL <http://dx.doi.org/10.1126/science.aao1733>.
- Noam Brown, Tuomas Sandholm. Superhuman ai for multiplayer poker. *Science*, 365(6456):885–890, 2019. doi: 10.1126/science.aay2400. URL <https://www.science.org/doi/abs/10.1126/science.aay2400>.
- Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder et al. Genie: Generative interactive environments. In *Forty-first International Conference on Machine Learning*, 2024.

- Sebastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke et al. Sparks of artificial general intelligence: Early experiments with gpt-4, 2023. URL <https://arxiv.org/abs/2303.12712>.
- Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker et al. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision, 2023. URL <https://arxiv.org/abs/2312.09390>.
- Keith T Butler, Daniel W Davies, Hugh Cartwright, Olexandr Isayev et al. Machine learning for molecular and materials science. *Nature*, 559(7715):547–555, 2018. doi: 10.1038/s41586-018-0337-2.
- Murray Campbell, A Joseph Hoane Jr, Feng-hsiung Hsu. Deep blue. *Artificial intelligence*, 134(1-2): 57–83, 2002. doi: 10.1016/s0004-3702(01)00129-1.
- Giuseppe Carleo, Ignacio Cirac, Kyle Cranmer, Laurent Daudet et al. Machine learning and the physical sciences. *Reviews of Modern Physics*, 91(4), 12 2019. ISSN 1539-0756. doi: 10.1103/revmodphys.91.045002. URL <http://dx.doi.org/10.1103/RevModPhys.91.045002>.
- Robin Ranjit Singh Chauhan. Martin Riedmiller. TalkRL: The Reinforcement Learning Podcast, 8 2023. URL <https://www.talkrl.com/episodes/martin-riedmiller>. Accessed: 2025-11-18.
- Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan et al. Evaluating large language models trained on code, 2021. URL <https://arxiv.org/abs/2107.03374>.
- Brian Christian. *The Alignment Problem: Machine Learning and Human Values*. W. W. Norton & Company, 2020. ISBN 978-0393635829.
- Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic et al. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.
- Casey Chu, Andrey Zhmoginov, Mark Sandler. CycleGAN, a master of steganography, 2017. URL <https://arxiv.org/abs/1712.02950>.
- Jack Clark, Dario Amodei. Faulty reward functions in the wild. OpenAI blog, 12 2016. URL <https://openai.com/index/faulty-reward-functions/>. Accessed: 2025-11-18.
- Jeff Clune. Humans and ai having convergent evolution of ideas. Twitter (X), December 17 2024. URL <https://x.com/jeffclune/status/1869134032899846293>.
- Cédric Colas, Tristan Karch, Olivier Sigaud, Pierre-Yves Oudeyer. Autotelic agents with intrinsically motivated goal-conditioned reinforcement learning: a short survey. *Journal of Artificial Intelligence Research*, 74:1159–1199, 2022. doi: 10.1613/jair.1.13554.
- Gheorghe Comanici, Eric Bieber, Mike Schaeckermann, Ice Pasupat et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025. doi: 10.48550/arXiv.2507.06261.
- Andrew Critch, Stuart Russell. Tarsa: a taxonomy and analysis of societal-scale risks from ai, 2023. URL <https://arxiv.org/abs/2306.06924>.
- Brandon Cui, Andrei Lupu, Samuel Sokota, Hengyuan Hu et al. Adversarial diversity in hanabi. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=uLE3WF3-H_5.
- Antoine Cully, Jeff Clune, Danesh Tarapore, Jean-Baptiste Mouret. Robots that can adapt like animals. *Nature*, 521(7553):503–507, May 2015. ISSN 1476-4687. doi: 10.1038/nature14422. URL <http://dx.doi.org/10.1038/nature14422>.
- DeepSeek-AI. Deepseek-v3 technical report, 2024. URL <https://arxiv.org/abs/2412.19437>.
- Jonas Degraeve, Federico Felici, Jonas Buchli, Michael Neunert et al. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897):414–419, 2 2022. ISSN 1476-4687. doi: 10.1038/s41586-021-04301-9. URL <http://dx.doi.org/10.1038/s41586-021-04301-9>.

- Branton DeMoss, Silvia Sapor, Jakob Foerster, Nick Hawes et al. The complexity dynamics of grokking. *Physica D: Nonlinear Phenomena*, 482:134859, 11 2025. ISSN 0167-2789. doi: 10.1016/j.physd.2025.134859. URL <http://dx.doi.org/10.1016/j.physd.2025.134859>.
- Michael Dennis, Natasha Jaques, Eugene Vinitzky, Alexandre Bayen et al. Emergent complexity and zero-shot transfer via unsupervised environment design. *Advances in neural information processing systems*, 33:13049–13061, 2020.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.
- Scott Lloyd DeWitt, Cynthia L. Selfe, Pamela Takayoshi. what video games have to teach us about learning and literacy. *College Composition and Communication*, 56(2):335–342, 2004. ISSN 0010096X. URL <http://www.jstor.org/stable/4140653>.
- Aaron Dharna, Amy K Hoover, Julian Togelius, Lisa B Soros. Transfer dynamics in emergent evolutionary curricula. *IEEE Transactions on Games*, 15(2):157–170, 2022. doi: 10.1109/tg.2022.3151025.
- Aaron Dharna, Cong Lu, Jeff Clune. Foundation model self-play: Open-ended strategy innovation via foundation models. *Reinforcement Learning Journal*, 6:276–342, 2025.
- Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O. Stanley et al. First return, then explore. *Nature*, 590(7847):580–586, 2 2021. ISSN 1476-4687. doi: 10.1038/s41586-020-03157-9. URL <http://dx.doi.org/10.1038/s41586-020-03157-9>.
- Scott Emmons, Erik Jenner, David K. Elson, Rif A. Saurous et al. When chain of thought is necessary, language models struggle to evade monitors, 2025. URL <https://arxiv.org/abs/2507.05246>.
- Jer Min Eyu, Kok-Lim Alvin Yau, Lei Liu, Yung-Wey Chong. Reinforcement learning in sentiment analysis: a review and future directions. *Artificial Intelligence Review*, 58(1), 11 2024. ISSN 1573-7462. doi: 10.1007/s10462-024-10967-0. URL <http://dx.doi.org/10.1007/s10462-024-10967-0>.
- Maxence Faldor, Jenny Zhang, Antoine Cully, Jeff Clune. Omni-epic: Open-endedness via models of human notions of interestingness with environments programmed in code. *arXiv preprint arXiv:2405.15568*, 2024. doi: 10.48550/arXiv.2405.15568.
- Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar et al. Minedojo: Building open-ended embodied agents with internet-scale knowledge. *Advances in Neural Information Processing Systems*, 35:18343–18362, 2022. doi: 10.52202/068431-1333.
- Lan-Tian Feng, Ming Zhang, Di Liu, Yu-Jie Cheng et al. On-chip quantum interference between the origins of a multi-photon state. *Optica*, 10(1):105–109, Jan 2023. doi: 10.1364/OPTICA.474750. URL <https://opg.optica.org/optica/abstract.cfm?URI=optica-10-1-105>.
- Tony Feng, Trieu Trinh, Garrett Bingham, Jiwon Kang et al. Semi-autonomous mathematics discovery with gemini: A case study on the erdos problems, 2026. URL <https://arxiv.org/abs/2601.22401>.
- Alexander Fleming. On the antibacterial action of cultures of a *Penicillium*, with special reference to their use in the isolation of *B. influenzae*. *British Journal of Experimental Pathology*, 10(3):226–236, June 1929. PMCID: PMC2048009.
- Jakob Foerster, Richard Y. Chen, Maruan Al-Shedivat, Shimon Whiteson et al. Learning with opponent-learning awareness. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS ’18, pp. 122–130, Richland, SC, 2018. International Foundation for Autonomous Agents and Multiagent Systems. doi: 10.65109/hgwa8807.
- Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick et al. Noisy networks for exploration. *arXiv preprint arXiv:1706.10295*, 2017. doi: 10.48550/arXiv.1706.10295.

- Lex Fridman. David silver: Alphago, alphazero, and deep reinforcement learning | lex fridman podcast #86, 2020. URL <https://www.youtube.com/watch?v=uPUeq8d73JI>.
- Iason Gabriel. Artificial intelligence, values, and alignment. *Minds and Machines*, 30(3):411–437, Sept 2020. ISSN 1572-8641. doi: 10.1007/s11023-020-09539-2. URL <http://dx.doi.org/10.1007/s11023-020-09539-2>.
- Javier García, Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(42):1437–1480, 2015. URL <http://jmlr.org/papers/v16/garcia15a.html>.
- A Gleave, M Dennis, N Kant, C Wild et al. Adversarial policies: Attacking deep reinforcement learning. *In Proc. ICLR-20*, 2020.
- Ian Goodfellow, Yoshua Bengio, Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu et al. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence et al. (eds.), *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/f033ed80deb0234979a61f95710dbe25-Paper.pdf.
- Charles A. E. Goodhart. Problems of monetary management: The UK experience. In *Monetary Theory and Practice*, pp. 91–121. Palgrave, London, 1984. doi: 10.1007/978-1-349-17295-5_4.
- Google DeepMind. AI achieves silver-medal standard solving international mathematical olympiad problems, 7 2024. URL <https://deepmind.google/discover/blog/ai-solves-imo-problems-at-silver-medal-level/>. Published 25 July 2024.
- A. N. Gorban, I. Y. Tyukin. Blessing of dimensionality: mathematical foundations of the statistical physics of data. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 376(2118):20170237, 3 2018. ISSN 1471-2962. doi: 10.1098/rsta.2017.0237. URL <http://dx.doi.org/10.1098/rsta.2017.0237>.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey et al. The llama 3 herd of models, 2024. URL <https://arxiv.org/abs/2407.21783>.
- Alex Graves. Generating sequences with recurrent neural networks, 2013. URL <https://arxiv.org/abs/1308.0850>.
- Alex Graves, Navdeep Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML’14, pp. II–1764–II–1772. JMLR.org, 2014.
- Ryan Greenblatt, Carson Denison, Benjamin Wright, Fabien Roger et al. Alignment faking in large language models, 2024. URL <https://arxiv.org/abs/2412.14093>.
- Melody Y. Guan, Miles Wang, Micah Carroll, Zehao Dou et al. Monitoring monitorability, 2025. URL <https://arxiv.org/abs/2512.18311>.
- William H. Guss, Mario Yncente Castro, Sam Devlin, Brandon Houghton et al. The minerl 2020 competition on sample efficient reinforcement learning using human priors, 2021a. URL <https://arxiv.org/abs/2101.11071>.
- William H. Guss, Cayden Codel, Katja Hofmann, Brandon Houghton et al. The minerl 2019 competition on sample efficient reinforcement learning using human priors, 2021b. URL <https://arxiv.org/abs/1904.10079>.

- William Hebgen Guss, Stephanie Milani, Nicholay Topin, Brandon Houghton et al. Towards robust and domain agnostic reinforcement learning competitions: Minerl 2020. In *NeurIPS 2020 Competition and Demonstration Track*, pp. 233–252. PMLR, 2021c.
- David Ha. Reinforcement learning for improving agent design. *Artificial Life*, 25(4):352–365, 11 2019. ISSN 1530-9185. doi: 10.1162/artl_a_00301. URL http://dx.doi.org/10.1162/artl_a_00301.
- David Ha, Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman et al. (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/2de5d16682c3c35007e4e92982f1a2ba-Paper.pdf.
- Dylan Hadfield-Menell, Stuart J Russell, Pieter Abbeel, Anca Dragan. Cooperative inverse reinforcement learning. *Advances in neural information processing systems*, 29, 2016.
- Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, Timothy Lillicrap. Mastering diverse domains through world models, 2024. URL <https://arxiv.org/abs/2301.04104>.
- Eric Hambro, Sharada Mohanty, Dmitrii Babaev, Minwoo Byeon et al. Insights from the neurips 2021 nethack challenge. In *NeurIPS 2021 Competitions and Demonstrations Track*, pp. 41–52. PMLR, 2022.
- Nikolaus Hansen. The cma evolution strategy: A tutorial, 2023. URL <https://arxiv.org/abs/1604.00772>.
- Dong Hao, Zhihai Rong, Tao Zhou. Extortion under uncertainty: Zero-determinant strategies in noisy games. *Physical Review E*, 91(5), 5 2015. ISSN 1550-2376. doi: 10.1103/PhysRevE.91.052803. URL <http://dx.doi.org/10.1103/PhysRevE.91.052803>.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016. doi: 10.1109/cvpr.2016.90.
- Erik Hemberg, Stephen Moskal, Una-May O’Reilly. Evolving code with a large language model. *Genetic Programming and Evolvable Machines*, 25(2):21, 2024. doi: 10.1007/s10710-024-09494-2.
- Geoffrey Hinton, Li Deng, Dong Yu, George E. Dahl et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012. doi: 10.1109/MSP.2012.2205597.
- Shogo Homma, Masanori Takezawa. Risk preference as an outcome of evolutionarily adaptive learning mechanisms: An evolutionary simulation under diverse risky environments. *PLOS ONE*, 19(8): e0307991, 8 2024. ISSN 1932-6203. doi: 10.1371/journal.pone.0307991. URL <http://dx.doi.org/10.1371/journal.pone.0307991>.
- Mia Hopman, Jannes Elstner, Maria Avramidou, Amritanshu Prasad et al. Evaluating and understanding scheming propensity in llm agents, 2026. URL <https://arxiv.org/abs/2603.01608>.
- Gregory Hornby, Al Globus, Derek Linden, Jason Lohn. Automated antenna design with evolutionary algorithms. In *Space 2006*. American Institute of Aeronautics and Astronautics, 9 2006. doi: 10.2514/6.2006-7242. URL <http://dx.doi.org/10.2514/6.2006-7242>.
- Shengran Hu, Jeff Clune. Thought cloning: Learning to think while acting by imitating human thinking. *Advances in Neural Information Processing Systems*, 36:44451–44469, 2023. doi: 10.52202/075280-1924.
- Shengran Hu, Cong Lu, Jeff Clune. Automated design of agentic systems. *arXiv preprint arXiv:2408.08435*, 2024. doi: 10.48550/arXiv.2408.08435.

- Thomas Hubert, Rishi Mehta, Laurent Sartran, Miklós Z. Horváth et al. Olympiad-level formal mathematical reasoning with reinforcement learning. *Nature*, 11 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09833-y. URL <http://dx.doi.org/10.1038/s41586-025-09833-y>.
- Evan Hubinger, Carson Denison, Jesse Mu, Mike Lambert et al. Sleeper agents: Training deceptive llms that persist through safety training, 2024. URL <https://arxiv.org/abs/2401.05566>.
- Sinan Ibrahim, Mostafa Mostafa, Ali Jnadi, Hadi Salloum et al. Comprehensive overview of reward engineering and shaping in advancing reinforcement learning applications. *IEEE Access*, 12:175473–175500, 2024. doi: 10.1109/access.2024.3504735.
- Imbue. Noam brown, fair: On achieving human-level performance in poker and diplomacy, and the power of spending compute at inference time, 2023. URL <https://imbue.com/podcast/2023-02-09-podcast-episode-27-noam-brown/>.
- Independent International Scientific Panel on Artificial Intelligence. Preliminary report of the independent international scientific panel on AI: Evidence-based assessment of opportunities, risks and impacts of AI. Technical report, United Nations, 7 2026. URL https://www.un.org/independent-international-scientific-panel-ai/sites/default/files/2026-07/en_Preliminary%20Report_.pdf.
- Dominique Jaccard, Laurent Suppan, Eric Sanchez, Audrey Huguenin et al. The co.lab generic framework for collaborative design of serious games: Development study. *JMIR Serious Games*, 9(3):e28674, Jul 2021. ISSN 2291-9279. doi: 10.2196/28674. URL <https://games.jmir.org/2021/3/e28674>.
- François Jacob. *The Possible and the Actual*. Jessie and John Danz Lectures. Pantheon Books, New York, 1982. ISBN 9780394706719.
- Daksh Jain, Aarya Jain, Ashutosh Desai, Avyakt Verma et al. Large language models as pokémon battle agents: Strategic play and content generation, 2025. URL <https://arxiv.org/abs/2512.17308>.
- Janus. Mysteries of mode collapse, 2022. <https://www.alignmentforum.org/posts/t9svvNPNmFf5Qa3TA/mysteries-of-mode-collapse>, Last accessed on 2025-09-17.
- Jiaming Ji, Tianyi Qiu, Boyuan Chen, Borong Zhang et al. Ai alignment: A comprehensive survey, 2025. URL <https://arxiv.org/abs/2310.19852>.
- John Jumper, Richard Evans, Alexander Pritzel, Tim Green et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021. doi: 10.1038/s41586-021-03819-2.
- Daniel Jurafsky, James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, with Language Models*. 3rd edition, 2026. URL <https://web.stanford.edu/~jurafsky/slp3/>. Online manuscript released January 6, 2026.
- Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan et al. Language models (mostly) know what they know, 2022. URL <https://arxiv.org/abs/2207.05221>.
- Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski et al. Model-based reinforcement learning for atari, 2024. URL <https://arxiv.org/abs/1903.00374>.
- Greg Kamradt. Needle in a haystack - pressure testing LLMs. GitHub repository, 2023. URL https://github.com/gkamradt/LLMTest_NeedleInAHaystack. Accessed: 2026-03-25.
- Seth Karten, Jake Grigsby, Tersoo Upaa Jr, Junik Bae et al. The pokeagent challenge: Competitive and long-context learning at scale, 2026. URL <https://arxiv.org/abs/2603.15563>.
- Diederik P Kingma, Max Welling. Auto-encoding variational bayes, 2013. URL <https://arxiv.org/abs/1312.6114>.

- Martin Klissarov, Pierluca D’Oro, Shagun Sodhani, Roberta Raileanu et al. Motif: Intrinsic motivation from artificial intelligence feedback. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=tmBKIEcDE9>.
- Tomek Korbak, Mikita Balesni, Elizabeth Barnes, Yoshua Bengio et al. Chain of thought monitorability: A new and fragile opportunity for ai safety. *arXiv preprint arXiv:2507.11473*, 2025. doi: 10.48550/arxiv.2507.11473.
- Raphael Koster, Jan Balaguer, Andrea Tacchetti, Ari Weinstein et al. Human-centred mechanism design with democratic ai. *Nature Human Behaviour*, 6(10):1398–1407, July 2022. ISSN 2397-3374. doi: 10.1038/s41562-022-01383-x. URL <http://dx.doi.org/10.1038/s41562-022-01383-x>.
- John R Koza. *Genetic programming*. Complex Adaptive Systems. Bradford Books, Cambridge, MA, 12 1992. ISBN 9780262111706.
- Victoria Krakovna, Jonathan Uesato, Vladimir Mikulik, Matthew Rahtz et al. Specification gaming: the flip side of ai ingenuity, 2020. <https://deepmind.google/discover/blog/specification-gaming-the-flip-side-of-ai-ingenuity/>, Last accessed on 2025-09-16.
- Mario Krenn, Mehul Malik, Robert Fickler, Radek Lapkiewicz et al. Automated search for new quantum experiments. *Phys. Rev. Lett.*, 116:090405, Mar 2016. doi: 10.1103/PhysRevLett.116.090405. URL <https://link.aps.org/doi/10.1103/PhysRevLett.116.090405>.
- Mario Krenn, Armin Hochrainer, Mayukh Lahiri, Anton Zeilinger. Entanglement by path identity. *Phys. Rev. Lett.*, 118:080401, Feb 2017. doi: 10.1103/PhysRevLett.118.080401. URL <https://link.aps.org/doi/10.1103/PhysRevLett.118.080401>.
- Mario Krenn, Jakob S Kottmann, Nora Tischler, Alán Aspuru-Guzik. Conceptual understanding through efficient automated design of quantum optical experiments. *Physical Review X*, 11(3):031044, 2021. doi: 10.1103/physrevx.11.031044.
- A. Krizhevsky, I. Sutskever, G. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pp. 1106–1114, 2012.
- Heinrich Küttler, Nantas Nardelli, Alexander Miller, Roberta Raileanu et al. The nethack learning environment. *Advances in Neural Information Processing Systems*, 33:7671–7684, 2020.
- Nathan Lambert. *Reinforcement Learning from Human Feedback*. Online, 2025. URL <https://rlhfbok.com>.
- Marc Lanctot, Vinicius Zambaldi, Audrunas Gruslys, Angeliki Lazaridou et al. A unified game-theoretic approach to multiagent reinforcement learning. *Advances in neural information processing systems*, 30, 2017.
- Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner et al. Measuring faithfulness in chain-of-thought reasoning, 2023. URL <https://arxiv.org/abs/2307.13702>.
- Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard et al. Rlaif vs. rlhf: Scaling reinforcement learning from human feedback with ai feedback, 2023.
- J. Lehman, K.O. Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation*, 19(2):189–223, 2011. doi: 10.1162/evco_a_00025.
- Joel Lehman, Jeff Clune, Dusan Misevic, Christoph Adami et al. The surprising creativity of digital evolution: A collection of anecdotes from the evolutionary computation and artificial life research communities. *Artificial life*, 26(2):274–306, 2020. doi: 10.1162/artl_a_00319.
- Joel Lehman, Jonathan Gordon, Shawn Jain, Kamal Ndousse et al. Evolution through large models. In *Handbook of evolutionary machine learning*, pp. 331–366. Springer, 2023.

- Joel Z. Leibo, Edward Hughes, Marc Lanctot, Thore Graepel. Autocurricula and the emergence of innovation from social interaction: A manifesto for multi-agent intelligence research, 2019. URL <https://arxiv.org/abs/1903.00742>.
- Jan Leike, David Krueger, Tom Everitt, Miljan Martic et al. Scalable agent alignment via reward modeling: a research direction. *arXiv preprint arXiv:1811.07871*, 2018. doi: 10.48550/arXiv.1811.07871.
- Sergey Levine, Chelsea Finn, Trevor Darrell, Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(39):1–40, 2016.
- Kevin Leyton-Brown, Yoav Shoham. *Essentials of Game Theory: A Concise, Multidisciplinary Introduction*. Morgan and Claypool Publishers, 1st edition, 2008. ISBN 1598295934.
- Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu et al. Code as policies: Language model programs for embodied control. In *arXiv preprint arXiv:2209.07753*, 2022. doi: 10.48550/arXiv.2209.07753.
- Chris Lu, Timon Willi, Christian A. Schroeder de Witt, Jakob N. Foerster. Model-free opponent shaping. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 14398–14411. PMLR, 2022.
- Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster et al. The AI Scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292*, 2024. doi: 10.48550/arXiv.2408.06292.
- Chris Lu, Cong Lu, Robert Tjarko Lange, Yutaro Yamada et al. Towards end-to-end automation of ai research. *Nature*, 651:914–919, 2026. doi: <https://doi.org/10.1038/s41586-026-10265-5>. URL <https://doi.org/10.1038/s41586-026-10265-5>.
- Jerry Luo, Cosmin Paduraru, Octavian Voicu, Yuri Chervonyi et al. Controlling commercial cooling systems using reinforcement learning, 2022. URL <https://arxiv.org/abs/2211.07357>.
- Aengus Lynch, Benjamin Wright, Caleb Larson, Kevin K. Troy et al. Agentic misalignment: How llms could be an insider threat. *Anthropic Research*, 2025. <https://www.anthropic.com/research/agentic-misalignment>.
- Yichuan Ma, Linyang Li, Yongkang Chen, Peiji Li et al. Mixing expert knowledge: Bring human thoughts back to the game of go, 2026. URL <https://arxiv.org/abs/2601.16447>.
- Alexander Meinke, Bronson Schoen, Jérémy Scheurer, Mikita Balesni et al. Frontier models are capable of in-context scheming, 2025. URL <https://arxiv.org/abs/2412.04984>.
- METR. Update on arc’s recent eval efforts. <https://metr.org/blog/2023-03-18-update-on-recent-evals/>, 03 2023.
- Cade Metz. How could AI destroy humanity? *The New York Times*, 6 2023. URL <https://www.nytimes.com/2023/06/10/technology/ai-humanity.html>. Accessed: 2023-06-10.
- Elliot Meyerson, Mark J. Nelson, Herbie Bradley, Adam Gaier et al. Language model crossover: Variation through few-shot prompting. *ACM Transactions on Evolutionary Learning and Optimization*, 4(4): 1–40, 11 2024. ISSN 2688-3007. doi: 10.1145/3694791. URL <http://dx.doi.org/10.1145/3694791>.
- Azalia Mirhoseini, Anna Goldie, Mustafa Yazgan, Joe Wenjie Jiang et al. A graph placement methodology for fast chip design. *Nature*, 594(7862):207–212, June 2021. ISSN 1476-4687. doi: 10.1038/s41586-021-03544-w. URL <http://dx.doi.org/10.1038/s41586-021-03544-w>.
- Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. doi: 10.1038/nature14236.
- Jean-Baptiste Mouret, Jeff Clune. Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*, 2015. doi: 10.48550/arXiv.1504.04909.

- Kevin P. Murphy. *Probabilistic Machine Learning: An introduction*. MIT Press, 2022. URL <http://probml.github.io/book1>.
- Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith et al. Progress measures for grokking via mechanistic interpretability. In *The Eleventh International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=9XFSbDPmdW>.
- Andrew Y. Ng, Daishi Harada, Stuart J. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, pp. 278–287, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc. ISBN 1558606122.
- Duy Nguyen-Tuong, Jan Peters, Matthias Seeger, Bernhard Schölkopf. Learning inverse dynamics: a comparison. In *Proceedings of the 16th European Symposium on Artificial Neural Networks (ESANN)*, pp. 13–18, 2008.
- Ben Norman, Jeff Clune. First-explore, then exploit: Meta-learning to solve hard exploration-exploitation trade-offs. *Advances in Neural Information Processing Systems*, 37:27490–27528, 2024. doi: 10.52202/079017-0864.
- Alexander Novikov, Ngăn Vũ, Marvin Eisenberger, Emilien Dupont et al. Alphaevolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025. doi: 10.48550/arXiv.2506.13131.
- OEIS Foundation Inc. Number of 1-factorizations of complete graph K_{2n} . The On-Line Encyclopedia of Integer Sequences, Entry A000438, 2026. URL <https://oeis.org/A000438>. Accessed: 2026-08-06.
- Open-Ended Team, Adam Stooke, Anuj Mahajan, Catarina Barros et al. Open-ended learning leads to generally capable agents. *arXiv preprint arXiv:2107.12808*, 2021. doi: 10.48550/arXiv.2107.12808.
- OpenAI. Introducing chatgpt. <https://openai.com/index/chatgpt/>, November 2022. Blog post.
- OpenAI. An openai model has disproved a central conjecture in discrete geometry. <https://openai.com/index/model-disproves-discrete-geometry-conjecture/>, 2026. Accessed: August 26, 2026.
- OpenAI, Christopher Berner, Greg Brockman, Brooke Chan et al. Dota 2 with large scale deep reinforcement learning, 2019. URL <https://arxiv.org/abs/1912.06680>.
- OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal et al. Gpt-4 technical report, 2024. URL <https://arxiv.org/abs/2303.08774>.
- OpenAI, :, Aaron Jaech, Adam Kalai et al. Openai o1 system card, 2026. URL <https://arxiv.org/abs/2412.16720>.
- Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022. doi: 10.52202/068431-2011.
- Alexander Pan, Kush Bhatia, Jacob Steinhardt. The effects of reward misspecification: Mapping and mitigating misaligned models, 2022. URL <https://arxiv.org/abs/2201.03544>.
- Joon Sung Park, Joseph O’Brien, Carrie Jun Cai, Meredith Ringel Morris et al. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023. doi: 10.1145/3586183.3606763.
- Jack Parker-Holder, Minqi Jiang, Michael Dennis, Mikayel Samvelyan et al. Evolving curricula with regret-based environment design. In *International Conference on Machine Learning*, pp. 17473–17498. PMLR, 2022.

- Jack Parker-Holder, Philip Ball, Jake Bruce, Vibhavari Dasagi et al. Genie 2: A large-scale foundation world model, 12 2024. URL <https://deepmind.google/discover/blog/genie-2-a-large-scale-foundation-world-model/>. Published 4 December 2024.
- Gian-Carlo Pascutto. Leela zero, 2019. URL <https://zero.sjeng.org>.
- Hammond Pearce, Baleegh Ahmad, Benjamin Tan, Brendan Dolan-Gavitt et al. Asleep at the keyboard? assessing the security of github copilot’s code contributions. *Communications of the ACM*, 68(2): 96–105, 2025. doi: 10.1145/3610721.
- Giuseppe Pellegrino. Sul massimo ordine delle calotte in $S_{4,3}$. *Le Matematiche*, 25:149–157, 1970. ISSN 0373-3505.
- Ian M. Pendleton, Gary Cattabriga, Zhi Li, Mansoor Ani Najeeb et al. Experiment specification, capture and laboratory automation technology (escalate): a software pipeline for automated chemical experimentation and data management. *MRS Communications*, 9(3):846–859, September 2019. ISSN 2159-6867. doi: 10.1557/mrc.2019.72. URL <http://dx.doi.org/10.1557/mrc.2019.72>.
- Ethan Perez, Sam Ringer, Kamile Lukošiušė, Karina Nguyen et al. Discovering language model behaviors with model-written evaluations. In *Findings of the association for computational linguistics: ACL 2023*, pp. 13387–13434, 2023. doi: 10.18653/v1/2023.findings-acl.847.
- Jan L Plass, Bruce D Homer, Charles K Kinzer. Foundations of game-based learning. *Educational psychologist*, 50(4):258–283, 2015. doi: 10.1080/00461520.2015.1122533.
- Marco Pleines, Daniel Addis, David Rubinstein, Frank Zimmer et al. Pokemon red via reinforcement learning. In *2025 IEEE Conference on Games (CoG)*, pp. 1–8. IEEE, 2025.
- Julien Pourcel, Cédric Colas, Gaia Molinaro, Pierre-Yves Oudeyer et al. Aces: Generating a diversity of challenging programming puzzles with autotelic generative models, 2024. URL <http://dx.doi.org/10.52202/079017-2160>.
- William H. Press, Freeman J. Dyson. Iterated prisoner’s dilemma contains strategies that dominate any evolutionary opponent. *Proceedings of the National Academy of Sciences*, 109(26):10409–10413, 2012. doi: 10.1073/pnas.1206569109. URL <https://www.pnas.org/doi/abs/10.1073/pnas.1206569109>.
- Kaiyi Qian, Kai Wang, Leizhen Chen, Zhaohua Hou et al. Multiphoton non-local quantum interference controlled by an undetected photon. *Nature Communications*, 14(1), 3 2023. ISSN 2041-1723. doi: 10.1038/s41467-023-37228-y. URL <http://dx.doi.org/10.1038/s41467-023-37228-y>.
- Linlu Qiu, Fei Sha, Kelsey Allen, Yoon Kim et al. Bayesian teaching enables probabilistic reasoning in large language models. *Nature Communications*, 17(1), 1 2026. ISSN 2041-1723. doi: 10.1038/s41467-025-67998-6. URL <http://dx.doi.org/10.1038/s41467-025-67998-6>.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning et al. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023. doi: 10.52202/075280-2338.
- Subhey Sadi Rahman, Md. Adnanul Islam, Md. Mahbub Alam, Musarrat Zeba et al. Hallucination to truth: a review of fact-checking and factuality evaluation in large language models. *Artificial Intelligence Review*, 59(2), 1 2026. ISSN 1573-7462. doi: 10.1007/s10462-025-11454-w. URL <http://dx.doi.org/10.1007/s10462-025-11454-w>.
- Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog et al. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 12 2023. ISSN 1476-4687. doi: 10.1038/s41586-023-06924-6. URL <http://dx.doi.org/10.1038/s41586-023-06924-6>.

- Stéphane Ross, Geoffrey Gordon, Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.
- Klaus F. Roth. On certain sets of integers. *Journal of the London Mathematical Society*, 1(1):104–109, 1953. doi: 10.1112/jlms/s1-28.1.104.
- David Rubinstein, Keelan Donovan, Daniel Addis, Kyoung Whan Choe et al. Learning Pokémon with Reinforcement Learning. <https://drubinstein.github.io/poker1/>, 2025. Accessed: August 26, 2026.
- Carlos Ruiz-Gonzalez, Sören Arlt, Jan Petermann, Sharareh Sayyad et al. Digital discovery of 100 diverse quantum experiments with pytheus. *Quantum*, 7:1204, 12 2023. ISSN 2521-327X. doi: 10.22331/q-2023-12-12-1204. URL <http://dx.doi.org/10.22331/q-2023-12-12-1204>.
- Sakana AI. The AI scientist: Towards fully automated open-ended scientific discovery. <https://sakana.ai/ai-scientist/>, August 2024. Accessed: 2026-05-08.
- Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar et al. The starcraft multi-agent challenge, 2019. URL <https://arxiv.org/abs/1902.04043>.
- Mikayel Samvelyan, Sharath C Raparthy, Andrei Lupu, Eric Hambro et al. Rainbow teaming: Open-ended generation of diverse adversarial prompts. *Advances in Neural Information Processing Systems*, 37:69747–69786, 2024. doi: 10.52202/079017-2229.
- Kumara Sastry, David E. Goldberg, Graham Kendall. Genetic algorithms. In Edmund K. Burke, Graham Kendall (eds.), *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, pp. 97–125. Springer, 2005. doi: 10.1007/0-387-28356-0_4. URL https://doi.org/10.1007/0-387-28356-0_4.
- Tom Schaul, Julian Togelius, Jürgen Schmidhuber. Measuring intelligence through games, 2011. URL <https://arxiv.org/abs/1109.1314>.
- Jérémy Scheurer, Mikita Balesni, Marius Hobbhahn. Large language models can strategically deceive their users when put under pressure, 2024. URL <https://arxiv.org/abs/2311.07590>.
- Tristan K. Schuler, Chinthan Prasad, Georgiy Kiselev, Donald Sofge. Seasonal station-keeping of short duration high altitude balloons using deep reinforcement learning. In *2025 IEEE Aerospace Conference*, pp. 1–11. IEEE, March 2025. doi: 10.1109/aero63441.2025.11068667. URL <http://dx.doi.org/10.1109/AER063441.2025.11068667>.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford et al. Proximal policy optimization algorithms, 2017. URL <https://arxiv.org/abs/1707.06347>.
- Marwin HS Segler, Mike Preuss, Mark P Waller. Planning chemical syntheses with deep neural networks and symbolic ai. *Nature*, 555(7698):604–610, 2018. doi: 10.1038/nature25978.
- Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud et al. Towards understanding sycophancy in language models, 2025. URL <https://arxiv.org/abs/2310.13548>.
- Minkyu Shin, Jin Kim, Bas van Opheusden, Thomas L. Griffiths. Superhuman artificial intelligence can improve human decision-making by increasing novelty. *Proceedings of the National Academy of Sciences*, 120(12), 3 2023. ISSN 1091-6490. doi: 10.1073/pnas.2214840120. URL <http://dx.doi.org/10.1073/pnas.2214840120>.
- Chenglei Si, Diyi Yang, Tatsunori Hashimoto. Can llms generate novel research ideas? a large-scale human study with 100+ nlp researchers. In *International Conference on Learning Representations*, volume 2025, pp. 94003–94092, 2025.

- David Silver, Aja Huang, Chris J. Maddison, Arthur Guez et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 1 2016. ISSN 1476-4687. doi: 10.1038/nature16961. URL <http://dx.doi.org/10.1038/nature16961>.
- David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017. doi: 10.1038/nature24270.
- David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018. doi: 10.1126/science.aar6404.
- Karen Simonyan, Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. URL <https://arxiv.org/abs/1409.1556>.
- Karl Sims. Evolving virtual creatures. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques - SIGGRAPH '94*, SIGGRAPH '94, pp. 15–22. ACM Press, 1994a. doi: 10.1145/192161.192167. URL <http://dx.doi.org/10.1145/192161.192167>.
- Karl Sims. Evolving 3d morphology and behavior by competition. *Artificial Life*, 1(4):353–372, 07 1994b. ISSN 1064-5462. doi: 10.1162/artl.1994.1.4.353. URL <https://doi.org/10.1162/artl.1994.1.4.353>.
- Avi Singh, Larry Yang, Kristian Hartikainen, Chelsea Finn et al. End-to-end robotic reinforcement learning without reward engineering, 2019. URL <https://arxiv.org/abs/1904.07854>.
- Joar Skalse, Nikolaus H. R. Howe, Dmitrii Krasheninnikov, David Krueger. Defining and characterizing reward hacking. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088. doi: 10.52202/068431-0687.
- Nat Sothanaphan. Resolution of erdos problem #728: a writeup of aristotle’s lean proof, 2026. URL <https://arxiv.org/abs/2601.07421>.
- Kenneth O Stanley, Joel Lehman. *Why greatness cannot be planned: The myth of the objective*. Springer, 2015. doi: 10.1007/978-3-319-15524-1.
- Alexander J. Stewart, Joshua B. Plotkin. From extortion to generosity, evolution in the iterated prisoner’s dilemma. *Proceedings of the National Academy of Sciences*, 110(38):15348–15353, 9 2013. ISSN 1091-6490. doi: 10.1073/pnas.1306246110. URL <http://dx.doi.org/10.1073/pnas.1306246110>.
- Yi Su, Dian Yu, Linfeng Song, Juntao Li et al. Crossing the reward bridge: Expanding rl with verifiable rewards across diverse domains, 2025. URL <https://arxiv.org/abs/2503.23829>.
- Jianwen Sun, Tianwei Zhang, Xiaofei Xie, Lei Ma et al. Stealthy and efficient adversarial attacks against deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 5883–5891, 2020. doi: 10.1609/aaai.v34i04.6047.
- Zhiqing Sun, Longhui Yu, Yikang Shen, Weiyang Liu et al. Easy-to-hard generalization: Scalable alignment beyond human supervision. *Advances in Neural Information Processing Systems*, 37: 51118–51168, 2024. doi: 10.52202/079017-1618.
- Ilya Sutskever, Jan Leike. Introducing superalignment. <https://openai.com/index/introducing-superalignment/>, July 2023. OpenAI Blog.
- Richard S Sutton, Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
- Nathan J Szymanski, Bernardus Rendy, Yuxing Fei, Rishi E Kumar et al. An autonomous laboratory for the accelerated synthesis of inorganic materials. *Nature*, 624(7990):86–91, 2023. doi: 10.1038/s41586-023-06734-w.

- Terence Tao. A digestion of the Jacobian conjecture counterexample, 7 2026. URL <https://terrytao.wordpress.com/2026/07/21/a-digestion-of-the-jacobian-conjecture-counterexample/>. Blog post, July 21, 2026.
- Emanuel Todorov, Tom Erez, Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012. doi: 10.1109/IROS.2012.6386109.
- Miles Turpin, Julian Michael, Ethan Perez, Samuel R. Bowman. Language models don’t always say what they think: Unfaithful explanations in chain-of-thought prompting, 2023. URL <https://arxiv.org/abs/2305.04388>.
- Teun Van Der Weij, Felix Hofstätter, Oliver Jaffe, Samuel Brown et al. Ai sandbagging: Language models can strategically underperform on evaluations. In *International Conference on Learning Representations*, volume 2025, pp. 73152–73189, 2025.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit et al. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. doi: 10.65215/ysbyhc05.
- Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michael Mathieu et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 10 2019. ISSN 1476-4687. doi: 10.1038/s41586-019-1724-z. URL <http://dx.doi.org/10.1038/s41586-019-1724-z>.
- ML Walker, DA Humphreys. Valid coordinate systems for linearized plasma shape response models in tokamaks. *Fusion Science and Technology*, 50(4):473–489, 2006. doi: 10.13182/fst06-a1271.
- Hanchen Wang, Tianfan Fu, Yuanqi Du, Wenhao Gao et al. Scientific discovery in the age of artificial intelligence. *Nature*, 620(7972):47–60, 8 2023a. ISSN 1476-4687. doi: 10.1038/s41586-023-06221-2. URL <http://dx.doi.org/10.1038/s41586-023-06221-2>.
- Rui Wang, Joel Lehman, Jeff Clune, Kenneth O. Stanley. Poet: open-ended coevolution of environments and their optimized solutions. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO ’19*, pp. 142–151, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450361118. doi: 10.1145/3321707.3321799.
- Rui Wang, Joel Lehman, Aditya Rawal, Jiale Zhi et al. Enhanced poet: Open-ended reinforcement learning through unbounded invention of learning challenges and their solutions. In *International Conference on Machine Learning*, pp. 9940–9951. PMLR, 2020.
- Tony Tong Wang, Adam Gleave, Tom Tseng, Nora Belrose et al. Adversarial policies beat superhuman go ais. In *International Conference on Machine Learning*, 2023b.
- Manuel Watter, Jost Tobias Springenberg, Joschka Boedecker, Martin Riedmiller. Embed to control: a locally linear latent dynamics model for control from raw images. In *Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 2, NIPS’15*, pp. 2746–2754, Cambridge, MA, USA, 2015. MIT Press.
- Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel et al. Emergent abilities of large language models, 2022a. URL <https://arxiv.org/abs/2206.07682>.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35: 24824–24837, 2022b. doi: 10.52202/068431-1800.
- Jerry Wei, Da Huang, Yifeng Lu, Denny Zhou et al. Simple synthetic data reduces sycophancy in large language models, 2024. URL <https://arxiv.org/abs/2308.03958>.
- Xumeng Wen, Zihan Liu, Shun Zheng, Shengyu Ye et al. Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base llms, 2025. URL <https://arxiv.org/abs/2506.14245>.

- John Wesson, D. J. Campbell. *Tokamaks*, volume 149 of *International Series of Monographs on Physics*. Oxford University Press, Oxford, UK, 4th edition, 2011. ISBN 9780199592234.
- Peter Whidden. Pokemonredexperiments. <https://github.com/PWhiddy/PokemonRedExperiments>, 2024.
- C.O. Wilke, J.L. Wang, C. Ofria, R.E. Lenski et al. Evolution of digital organisms at high mutation rates leads to survival of the flattest. *Nature*, 412(6844):331–333, 2001. doi: 10.1038/35085569.
- Peter R. Wurman, Samuel Barrett, Kenta Kawamoto, James MacGlashan et al. Outracing champion gran turismo drivers with deep reinforcement learning. *Nature*, 602(7896):223–228, 2 2022. ISSN 1476-4687. doi: 10.1038/s41586-021-04357-7. URL <http://dx.doi.org/10.1038/s41586-021-04357-7>.
- Yutaro Yamada, Robert Tjarko Lange, Cong Lu, Shengran Hu et al. The ai scientist-v2: Workshop-level automated scientific discovery via agentic tree search, 2025. URL <https://arxiv.org/abs/2504.08066>.
- Kaiyu Yang, Aidan M. Swope, Alex Gu, Rahul Chalamala et al. Leandojo: Theorem proving with retrieval-augmented language models, 2023. URL <https://arxiv.org/abs/2306.15626>.
- Georgios N. Yannakakis, Julian Togelius. *Artificial Intelligence and Games*. Springer Nature, 2 edition, 2025. doi: <http://dx.doi.org/10.1007/978-3-031-83347-2>. <https://gameaibook.org>.
- Jason Yosinski, Jeff Clune, Yoshua Bengio, Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014.
- Yang Yu. Towards sample efficient reinforcement learning. In *IJCAI*, pp. 5739–5743, 2018. doi: 10.24963/ijcai.2018/820.
- Fuxiang Zhang, Junyou Li, Yi-Chen Li, Zongzhang Zhang et al. Improving sample efficiency of reinforcement learning with background knowledge from large language models. *IEEE Transactions on Neural Networks and Learning Systems*, 2025. doi: 10.1109/tnnls.2025.3590731.
- Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange et al. Darwin godel machine: Open-ended evolution of self-improving agents, 2026a. URL <https://arxiv.org/abs/2505.22954>.
- Jenny Zhang, Bingchen Zhao, Wannan Yang, Jakob Foerster et al. Hyperagents, 2026b. URL <https://arxiv.org/abs/2603.19461>.
- Ruixun Zhang, Thomas J. Brennan, Andrew W. Lo. The origin of risk aversion. *Proceedings of the National Academy of Sciences*, 111(50):17777–17782, 12 2014. ISSN 1091-6490. doi: 10.1073/pnas.1406755111. URL <http://dx.doi.org/10.1073/pnas.1406755111>.
- Tianjun Zhang, Huazhe Xu, Xiaolong Wang, Yi Wu et al. Bebold: Exploration beyond the boundary of explored regions, 2020. URL <https://arxiv.org/abs/2012.08621>.
- Victor Zhong, Austin W. Hanjie, Sida I. Wang, Karthik Narasimhan et al. Silg: The multi-environment symbolic interactive language grounding benchmark, 2022. URL <https://arxiv.org/abs/2110.10661>.
- Jun-Yan Zhu, Taesung Park, Phillip Isola, Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017. doi: 10.1109/iccv.2017.244.
- Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pp. 928–936, 2003.
- X. Y. Zou, L. J. Wang, L. Mandel. Induced coherence and indistinguishability in optical interference. *Physical Review Letters*, 67(3):318–321, 7 1991. ISSN 0031-9007. doi: 10.1103/physrevlett.67.318. URL <http://dx.doi.org/10.1103/PhysRevLett.67.318>.

Supplementary Material

Table of Contents

8	Breakdown of Story Attribution	57
8.1	Call for Anecdotes	57
8.2	List of Anecdotes	59

8 Breakdown of Story Attribution

8.1 Call for Anecdotes

Below is the call for anecdotes we put out into our various networks and public social media sites. In addition to posting to our various networks, we reached out to some researchers directly if we believed there was a chance that they had a story that would fit this collection or if they had previously shared such a story with us, e.g., over drinks at a conference:

Dear colleagues,

TL;DR: Please submit (to aifindsaway@gmail.com) any stories you know of where AI acted in a way that surprised its creators, especially if it could be seen as unsafe (e.g. hacking a reward function, finding a loophole in an environment or experimental design, goal misgeneralization, etc.).

As AI researchers, we know that AI is creative and constantly surprises us, often outwitting our experimental designs and forcing us to iterate to close loopholes on things like reward functions and environment configurations. To anthropomorphize, it can seem mischievous or clever at times. These stories are important as society grapples with the question of AI Safety and Existential Risk, as they teach us how the unexpected is routine, and how we often fail to anticipate ways in which AI will escape our attempts to contain it. Such stories thus inform scientists, the general public, and regulators. However, these important anecdotes are usually passed around orally, meaning we do not know to what extent they are true, and it is hard for scientists and regulators to include them in official documents. To remedy these issues, we aim to record the true accounts of as many anecdotes as possible regarding AI (of any type, including RL, ML, etc.) surprising its creators and users.

This effort (by Aaron Dharna, Cong Lu, Joel Lehman, Victoria Krakovna, and Jeff Clune) is a follow-up to our 2018 paper *The Surprising Creativity of Digital Evolution* (TSCDE). That paper is a crowdsourced collection of anecdotes from the artificial life and evolutionary computation communities about how their algorithms creatively subverted expectations. That paper made an important contribution to ongoing discussions of AI Safety, but was limited because its scope was confined to one narrow area of AI (evolutionary methods). Our new paper expands the scope to all areas of AI, especially the most powerful methods (deep learning, including deep reinforcement learning).

This expansion of previous work is driven in part by the response of the AI Safety community to TSCDE, which has become a valuable resource for them in publications [1, 2, 3] and in teaching future leaders in AI safety by being included in AI safety course syllabi (e.g. UC Berkeley’s Safety and Control for Artificial General Intelligence course).

Please send us any accounts you think we should include. If we add it to the paper, the scientists involved will be appropriately cited and/or mentioned to give them credit. Unfortunately, we ran into many challenges throughout the process with TSCDE with submitters as co-authors, so this time we are instead recognizing contributors by name and with all appropriate citations in the paper. If you know of an account but did not perform the experiment yourself, please tell us what you know, including who we might contact for a firsthand account.

We hope you can help create an account of these fascinating and sometimes ominous anecdotes so we can inform AI safety discussions, either by submitting and/or spreading the word of this Call for Anecdotes.

More details below.

Thanks,

Aaron, Cong, Joel, Victoria, and Jeff

Please send us a quick summary of your anecdote. We can then let you know if we will include it in the paper, at which point we may ask for more details.

An example anecdote is available here, which can serve as a rough guide to the length, level of detail, and surprise factor we are looking for. Please copy the document and use it as a template for submissions. We will curate and edit these into a full publication. Before the camera-ready publication is released, we will provide you with the opportunity to read the paper and make sure you are happy with your contribution.

Please forward this email to whomever you think might have an interesting anecdote to share. We look forward to your exciting, amusing, worrisome, and/or insightful contributions!

References

- [1] Jess Whittlestone, Kai Arulkumaran, and Matthew Crosby. “The Societal Implications of Deep Reinforcement Learning”. In: *Journal of Artificial Intelligence Research* 70 (Mar. 2021). issn: 1076-9757. doi: 10.1613/jair.1.12360. url: <http://dx.doi.org/10.1613/jair.1.12360>.
- [2] Tom Everitt, Gary Lea, and Marcus Hutter. “AGI Safety Literature Review”. In: *IJCAI’18*. Stockholm, Sweden: AAAI Press, 2018, pp. 5441-5449. isbn: 9780999241127
- [3] Robert Geirhos et al. “Shortcut learning in deep neural networks”. In: *Nature Machine Intelligence* 2.11 (Nov. 2020), pp. 665-673. issn: 2522-5839. doi:10.1038/s42256-020-00257-z

8.2 List of Anecdotes

10 of the anecdotes come from the public record. The remaining 16 are new to this collection.

- Section 3.1: David Silver’s section comes from his interview with Lex Fridman (Fridman, 2020); Marc Lanctot’s is new; therefore, we count this as both a new addition and an item pulled from the public record.
- Section 3.2: comes from Noam Brown’s interview with Kanjun Qiu at Imbue (Imbue, 2023)
- Section 3.3: new
- Section 3.4: new
- Section 4.1.1: comes from the OpenAI blog post (Clark & Amodei, 2016)
- Section 4.1.2: new
- Section 4.1.3: comes from the OpenAI blog post (Amodei et al., 2017)
- Section 4.1.4: details of the reward hacking/collusion between agents are new to this work, but the experimental setup is described in Dennis et al. (2020)
- Section 4.1.5: new
- Section 4.1.6: new
- Section 4.2.1: comes from the OpenAI blog post (Baker et al., 2020) and Jeff Clune
- Section 4.2.2: new
- Section 4.2.3: comes from David Ha’s blog post version of the published paper about World Models (Ha & Schmidhuber, 2018)
- Section 4.2.4: new
- Section 4.2.5: comes from Bird & Layzell (2002)
- Section 4.2.6: comes from OpenAI’s technical report on GPT-4 (METR, 2023; OpenAI et al., 2024)
- Section 5.1: new
- Section 5.2: new
- Section 5.3: new
- Section 5.4: new
- Section 5.5: comes from a blog post and responding comment from Paul Christiano (Janus, 2022)
- Section 5.6: comes from the Twitter post (Albert, 2024)
- Section 6.1: comes from Riedmiller’s interview on the TalkRL podcast (Chauhan, 2023)
- Section 6.2: some details are described in Romera-Paredes et al. (2023), but new details were provided by Alex Novikov as well, so this counts as new
- Section 6.3: some details were described in Sakana AI (2024), but Cong Lu provided new details too, so this counts as new
- Section 6.4: new