ROYAL SOCIETY
OPEN SCIENCE

royalsocietypublishing.org/journal/rsos

Research

Cite this article: Jiang M, Rocktäschel T, Grefenstette E. 2023 General intelligence requires rethinking exploration. *R. Soc. Open Sci.* **10**: 230539.
https://doi.org/10.1098/rsos.230539

Received: 24 April 2023
Accepted: 30 May 2023

Subject Category:
Computer science and artificial intelligence

Subject Areas:
artificial intelligence

Keywords:
artificial intelligence, machine learning, exploration, data curation

Authors for correspondence:
Minqi Jiang
e-mail: mnqjng@gmail.com
Edward Grefenstette
e-mail: edward@egrefen.com

General intelligence requires rethinking exploration

Minqi Jiang, Tim Rocktäschel and Edward Grefenstette

AI Centre, Department of Computer Science, University College London, London, UK

EG, 0000-0003-1164-8809

We are at the cusp of a transition from 'learning from data' to 'learning what data to learn from' as a central focus of artificial intelligence (AI) research. While the first-order learning problem is not completely solved, large models under unified architectures, such as transformers, have shifted the learning bottleneck from how to effectively train models to how to effectively acquire and use task-relevant data. This problem, which we frame as *exploration*, is a universal aspect of learning in open-ended domains like the real world. Although the study of exploration in AI is largely limited to the field of reinforcement learning, we argue that exploration is essential to all learning systems, including supervised learning. We propose the problem of *generalized exploration* to conceptually unify exploration-driven learning between supervised learning and reinforcement learning, allowing us to highlight key similarities across learning settings and open research challenges. Importantly, generalized exploration is a necessary objective for maintaining open-ended learning processes, which in continually learning to discover and solve new problems, provides a promising path to more general intelligence.

1. Introduction

A hallmark of intelligence is a capacity to explore. From animals exploring their surroundings for food and shelter [1–3] to children seeking novelty in play [4–9], exploration drives the acquisition of new information beneficial to the seeker. Exploration is even coded in the very fabric of life, in the form of genetic mutations that wander the space of phenotypes—some of which may improve the organism's chances to survive and reproduce. Through exploration, the explorer acquires not just information about a specific task or environment in isolation, but information in relation to previous experiences, informing the development of more generally useful behaviours—which may include general strategies for searching for rewarding states in similar environments. Still, such information acquisition is not purely driven by the explorer. It is also determined by the learning opportunities provided by the environment: a static environment

© 2023 The Authors. Published by the Royal Society under the terms of the Creative Commons Attribution License http://creativecommons.org/licenses/by/4.0/, which permits unrestricted use, provided the original author and source are credited.

THE ROYAL SOCIETY
PUBLISHING

presenting a fixed challenge that the agent can already solve offers nothing more to learn. Similarly, a variable environment for which the agent already learned to explore optimally to find the solution offers no potential for learning new behaviours. Conversely, an environment presenting challenges far exceeding the capabilities of the agent is unlikely to provide sufficiently informative experiences for the agent to learn. Exploration should thus seek information most useful for the agent to become a more general and adept actor in future decision-making situations. In this way, exploration serves as the core mechanism for generating new experiences from which agents can learn to extrapolate or rapidly adapt to a changing world—to become a more generally intelligent agent within the world.

The subject of generality in intelligence is a loaded concept to say the least. In broaching this subject, we must first clarify our treatment of general intelligence in the context of our discussion. As the set of tasks to which the notion of generality pertains is typically either undefined or infinite, definitions of general intelligence tend to be vague, incomputable or unquantifiable in degree. Nevertheless, the quest for producing ever more general models has driven much progress in AI research [10–14]. To avoid these definitional issues, we instead focus on a relative notion of general intelligence: model A is more general than model B relative to a task set $\mathcal{T}$ if and only if A performs above a threshold level (e.g. that of a minimally viable solution) in more tasks in $\mathcal{T}$ than B, while at least matching the performance of B on all other tasks in $\mathcal{T}$ on which B meets the threshold. Under this definition, general intelligence is not necessarily the end state of any system, but rather a property that can change over time, relative to other intelligent systems and the specific task domain. We can then refer to a system exhibiting continual improvements in relative general intelligence as an *increasingly general intelligence* (IGI). Over time an IGI will overtake any other agent that is not an IGI in terms of relative general intelligence. For these reasons, our discussion conceives of ‘general intelligence’ as an IGI. In other words, we equate the goal of producing a general intelligence with producing an IGI. Moreover, for brevity, we exclude explicit reference to the associated task set on which IGI is defined. If not obvious given the context, the reader can assume we refer to the set of tasks which humans engage in, would engage in, or would benefit from. Further, we do not aim to address whether the notion of a general intelligence relative to the space of all possible tasks is well defined, as it is an orthogonal concern to the argument we lay out in this paper.

At a high level, a learning process leading to general intelligence must explore across two levels of abstraction: first, exploration within specific environments facilitates the search for optimal solutions. Second, exploration across environments ensures continual learning progress and developing more general capabilities. If we assume the agent retains past solutions and continues to discover and explore new challenges indefinitely, we can expect the agent to attain a broad spectrum of capabilities [10]. Importantly, its capabilities should eventually surpass those obtained by any process with less effective exploration. Inversely, if the agent does not explore as thoroughly, we cannot expect it to acquire more capabilities than if it did. Therefore, increasingly general intelligence arises if and only if facilitated by open-ended exploration, which continually broadens the agent’s capabilities. Such exploration ideally seeks new challenges at the *boundary* of the agent’s capabilities, focusing on tasks for which the current agent has the most potential to improve—similar to concepts from developmental psychology, like Vygotsky’s Zone of Proximal Development (ZPD) [15]. Here, the exploratory process can be considered a separate ‘teacher’ component that guides the ‘student’ IGI toward tasks at the boundary of the IGI’s capabilities. When expert demonstrations are available to expedite learning via imitation, this learning process directly resembles the traditional ZPD setting. Likely, such demonstrations are unavailable, and the exploratory process provides ‘scaffolding,’ whereby the teacher shapes the structure of the task to promote the student’s learning progress [16].

In sharp contrast, the current research programs producing state-of-the-art machine learning (ML) systems, including large language-models (LLMs) [12,17,18] and generative models of images and video [19–21], strive for generality without exploration. These large deep learning models are typically trained using variations of supervised learning (SL), including self-supervised learning [22], and benchmarked on static datasets collected entirely offline. We argue that this paradigm leads to models with unavoidable brittleness. Models trained in this way will exhibit blindspots reflective of the missing information in their training data and suffer from covariate shifts and concept drift due to the non-stationarity of the real world.

Meanwhile, deep reinforcement learning (RL), which seeks to learn sequential decision-making strategies in simulated or real environments, directly considers the exploration–exploitation trade-off, which treats exploration as a key objective, balanced alongside maximizing task-specific performance [23]. RL acknowledges that task performance must often suffer in the short term in service to exploration—often necessary for improved performance. Unlike SL models, which can take advantage of an effectively infinite trove of data on the open web, the state-of-the-art RL methods remain limited to largely toy simulations or highly domain-specific applications. RL agents are largely limited to training in simulation for a number of reasons, although most typically this is due to the potential costs (e.g. financial, temporal, environmental) and dangers of real-world experience collection. As we argue in §2.2, current simulators predominantly mirror the limitations of a static, finite dataset in SL, and as such, RL agents share the same failure points as SL systems when deployed in the wild. Throughout this paper, we will use the terms *static environment* or *static simulator* to refer to environments whose factors of variation form a constant set throughout the course of training. This definition encompasses both singleton environments, like each game in the popular Atari Learning Environment [24], which exist in the same configuration across every training episode, as well as environments like NetHack [25], which have a predefined set of configuration parameters whose values control specific attributes such as its appearance, topology, transition dynamics and reward functions. Such attributes may take on different values in each training episode. While a static simulator with adjustable parameters can span a vast space of tasks, ultimately, it can only offer experiences within the limited domain that it was designed to simulate.

To introduce a case that will serve as a running example throughout this paper, consider a virtual assistant that responds to user queries through a chat-based interface and performs actions on behalf of the user by interfacing with other software, e.g. answering questions by searching the web, finding and buying products, hailing a cab or providing music and movie recommendations. Such an assistant could, in principle, learn to execute any task that can be performed digitally, which is becoming the majority of tasks humans in the information age are performing. Our assistant has the potential to become increasingly generally intelligent, assuming it can sufficiently explore the space of tasks. Following our discussion, we see that current methods for training our assistant will not suffice: training an increasingly generally intelligent assistant requires continually exploring the vast and dynamically evolving space of tasks of interest to people, but current SL and RL methods are largely designed for training on a single, static dataset or simulator. An assistant trained with such approaches would quickly grow outdated with the onset of new-world events and product offerings. The subfield of continual learning [26] seeks to reliably retrain models on new data, but does not directly address the question of how to determine and find the data for training in the first place.

We see that the problem afflicting both classes of learning algorithms reduces to one of insufficient exploration: SL, largely trapped in the offline regime, fails to perform any exploration, while RL, limited to exploring the interior of a static simulation, largely ignores the greater expanse of possibilities that the simulation cannot express. Thus, simply applying the current exploration paradigms developed in RL to SL is not enough to solve these problems. In order to properly address these limitations, we must first rethink exploration as it is currently framed in RL. We require a more general kind of exploration, which searches beyond the confines of a static data generator, such as a finite dataset or static simulator. This generalized form of exploration must deliberately and continually seek out promising data to expand the learning agent’s repertoire of capabilities. Such expansion necessarily entails searching outside of what can be sampled from a static data generator. This open-ended exploration process, summarized in figure 1, defines a new data-seeking outer-loop that continually expands the data generator used by the inner loop learning process, which itself may use more limited forms of exploration to optimally sample from this data generator. Within each inner loop, the data generator is static, but as a whole, this open-ended exploration process defines a dynamic, adaptive search process that generates the data necessary for training a likewise open-ended learner that, over time, may attain increasingly general capabilities.

Importantly, we argue open-ended exploration is necessary for collecting the most informative training data to feed our ever more powerful and data-hungry machine learning models. Previous studies suggest that, while model performance grows with the amount of training data, the quality of the data itself is of paramount importance [17,27–29]. As our understanding of optimization, modelling, and scaling laws improve in tandem with the continued expansion of computing power following extensions of Moore’s Law [30,31], informative data is not only a limiting factor to more general models, but also an important counterweight to ensuring we do not overfit our models and algorithms to a limited subset of problems. Moreover, tailoring the sequence of training data presented to a model can greatly benefit sample efficiency and the quality of the learned representations. Continual exploration of informative data also serves as a critical means to ensure models remain well-aligned to an increasingly diverse world of dynamic human preferences and values. In this light, we argue a significant amount of research investment should shift toward the collection, design, and scheduling of training data itself—that is, toward developing

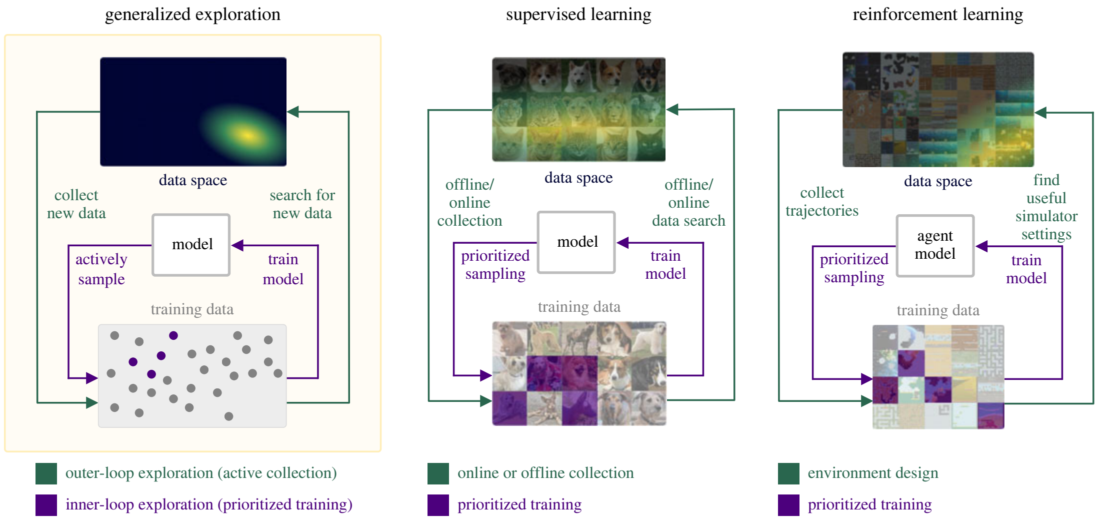

Figure 1. A general framework for exploration: an outer loop performs active collection of new training data, and an inner loop conducts prioritized training on the current training data. In SL, the outer loop consists of either online or offline data collection. In RL, the outer loop searches for simulator settings that yield useful training data, and the inner loop can perform prioritized sampling, e.g. prioritized experience replay.

principled methods for open-ended exploration. Combined with ever-improving model architectures and optimization techniques, such generalized exploration can lead to increasingly more general intelligent agents.

Many prior works propose the potential of open-ended self-improving systems. Notably, both Schmidhuber [10,32,33] and Clune [34] have proposed novelty-seeking agents that generate their own challenges, while the work of Stanley and Lehman established the practical value of taming open-endedness [35–37]. We do not claim to be the first to propose the concept. Rather, we seek to provide a detailed discussion of how open-ended learning can be integrated into modern ML systems and importantly, how this endeavour entails a meaningful generalization of prevailing notions of exploration. In the rest of this paper, we formalize the motivation and formulation of open-ended exploration. In §2 we first build up a detailed discussion about the challenges that must be addressed by a successful framework of exploration for both SL and RL. An especially important challenge that motivates the need for continual exploration is the bootstrap problem in online learning, which we discuss in §3. We then propose a unified framework for exploration in §4, generalizing its application across problem domains and thereby providing a common framework for thinking about data collection in both SL and RL. Section 5 provides a discussion of how such open-ended learning may take place in bath practice and in research. As such open-ended learning provides a promising path to general intelligence, we relate these ideas to other proposed approaches achieving this grandiose goal in §6. Finally, in §7, we discuss what we consider the key open problems to realizing this path to general intelligence.

2. Learning is not enough

Over the past decade rapid advances in model design and optimization have drastically improved our ability to train effective models in complex domains. Recent deep SL methods perform highly accurate modelling across modalities like natural language [12,18,20], images [19,38] and speech [39,40]. In parallel, progress in deep RL has produced agents matching or beating top human players in strategic games like Go [41,42] and Stratego [43], while achieving state-of-the-art results in scientific pursuits like chip design [44,45] and controlling nuclear fusion plasma [46]. While impressive, these developments focus primarily on questions of model design and optimization. There remain many open questions on which aspects of intelligence cannot be adequately captured by existing approaches, e.g. robust symbolic reasoning [47], causal understanding [48,49], hierarchical planning in a world model [50] and large-scale collective behaviours like cultural learning [51–53]. We argue that, even if provided a joint model and optimization scheme that captures these missing aspects of intelligence, the training data itself remains a fundamental limitation, and addressing it requires making progress on a host of separate problems. From this perspective, the historic focus on model design and optimization points to a systematic underinvestment in *exploration*—how the training data are collected or created in the first place, whether in the form of a dataset or simulator. The generality of both SL and RL models are ultimately bottlenecked on the diversity of this training data—a limitation demonstrated in many recent works [54–57]. RL has historically focused on exploration as a principal objective for efficient learning, and indeed these ideas hint at a path toward addressing the data limitations of model learning in general. In this section, we consider these limitations in SL and RL and argue that despite its focus on exploration, RL ultimately suffers from the same data limitations as SL. We propose that we must rethink the existing exploration paradigm in RL to go beyond the data bottleneck on the path to general intelligence.

2.1. The limits of supervised learning

Supervised learning aims to learn a function f mapping points in domain $\mathcal{X}$ to domain $\mathcal{Y}$, given N training examples $(x_i, y_i)_{i=0}^{N}$ reflecting this mapping, where $x \in \mathcal{X}$, $y \in \mathcal{Y}$. In modern ML, this function is typically a large artificial neural network with parameters denoted $\theta \in \Theta$. The defining feature of SL is that learning proceeds by optimizing a loss function $\mathcal{L} : \Theta \times \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ that provides the error between the model's prediction $f(x)$ and the true value y paired with x. In general, each training example (x, y) can be seen as a sample from a ground-truth distribution $P(x, y)$, because the true data generating function $f^* : \mathcal{X} \rightarrow \mathcal{Y}$ can be stochastic. Therefore, the goal of SL can be seen as learning an approximator $f : \mathcal{X} \rightarrow \mathcal{Y}$ that produces samples y consistent with $P(y \mid x)$, for example, by using a loss function that encourages f to deterministically predict the mean or mode of $P(y \mid x)$, or that matches the distribution of outputs of $f(x)$ to $P(y \mid x)$.

Crucially, this definition of SL assumes the model f is trained once on a static, finite training dataset, $\mathcal{D}_{\text{train}}$. We should thus not expect f to accurately model data that differs significantly from its training data. Nevertheless, with massive amounts of training data, large models can exhibit impressive generality [12,18,19] and recent scaling laws suggest that test performance should improve further with even more data [17,56]. Given the benefits of data scale, contemporary state-of-the-art SL models are trained on internet-scale, offline datasets, typically harvested via webcrawling. While such datasets may capture an impressive amount of information about the world, they inevitably fall short in containing all relevant information that a model may need when deployed in the wild. All finite, offline datasets share two key shortcomings: *incompleteness*, as the set of all facts about the world is infinite [58], and *stationarity*, as such datasets are by definition fixed. For example, our virtual assistant, if trained on a static conversational corpus, would soon see its predictions grow irrelevant, as its model falls out of date with culture, world events, and even language usage itself. Indeed, all ML systems deployed in an open-world setting, with real users and peers, must continually explore and train on new data, or risk fading into irrelevance. What data should the system designer (or the system itself) collect next for further training? This is the complementary—and equally important—problem of exploration that sits beneath all ML systems in deployment, one that has been considered at length in the field of RL.

2.2. The limits of reinforcement learning

RL is a general formulation of the problem of online learning. In RL, the decision-making model, or *agent*, interacts with its environment at each time t by observing an observation o_t emitted by the underlying Markov state s_t of the environment and taking an action a_t according to its *policy* $\pi(a_t \mid o_t)$. In response, the environment transitions to its next state s_{t+1} according to the transition function $\mathcal{T}(s_{t+1}|s_t, a_t)$ and the agent receives a reward r_{t+1} according to the reward function $\mathcal{R}(s_{t+1})$. Such an environment is called a Markov Decision Process (MDP). The policy π is typically parametrized as a deep neural network with parameters θ and optimized to maximize the expected future discounted return—that is the discount sum of future rewards—assuming some prior p over the initial state and a discount factor $\gamma \leq 1$:

$$J(\theta) = \mathbb{E}_{\pi}\left[\sum_{t} \gamma^t r_t\right]. \tag{2.1}$$

The expected future discounted return from a specific state s is also called the value of s. Here, the expectation over returns is over the specific distribution of states induced by following the policy π in the environment. RL maximizes equation (2.1) by solving the *credit assignment problem*, that is, estimating the future value of taking each action for each state of the world, thereby reducing the problem to a form of dynamic programming [59]: the optimal policy then follows from taking the highest value action in each state.

As RL agents train on their own experiences, locally optimal behaviours can easily self-reinforce, preventing the agent from reaching better optima. To avoid such outcomes and ensure sufficient coverage of possible MDP transitions during training, RL considers exploration a principal aim. Under exploration, the RL agent performs actions in order to maximize some measure of novelty of the resulting experiential data or uncertainty in outcome, rather than to maximize return. Simpler still, new and informative states can often be unlocked by injecting noise into the policy, e.g. by sporadically sampling actions uniformly at random. However, such random search strategies can run into the curse of dimensionality, becoming less sample efficient in practice.

A prominent limitation of state-of-the-art RL methods is their need for large amounts of data to learn optimal policies. This sample inefficiency is often attributable to the sparse-reward nature of many RL environments, where the agent only receives a reward signal upon performing some desired behaviour. Even in a dense reward setting, the agent may likewise see sample-inefficient learning once trapped in a local optimum, as discovering pockets of higher reward can be akin to finding a similarly sparse signal. In complex real-world domains with large state spaces and highly branching trajectories, finding the optimal behaviour may require an astronomical number of environment interactions, despite performing exploration. Thus for many tasks, training an RL agent using real-world interactions is highly costly, if not completely infeasible. Moreover, a poorly trained embodied agent acting in real-world environments can potentially perform unsafe interactions. For these reasons, RL is typically performed within a simulator, with which massive parallelization can achieve billions of samples within a few hours of training [60,61].

Simulation frees RL from the constraints of real-world training at the cost of the *sim2real gap*, the difference between the experiences available in the simulator and those in reality. When the sim2real gap is high, RL agents perform poorly in the real world, despite succeeding in simulation. Importantly, a simulator that only implements a single task or small variations thereof will not produce agents that transfer to the countless tasks of interest for general intelligence. Thus, RL ultimately runs into a similar data limitation as in SL. Our virtual assistant, trained to navigate the web in a static simulation, cannot be expected to generalize to the many potential variants of webpages, both current and future, which may incorporate design, interaction and structural paradigms completely missing in the original simulation. In fact, the situation may be orders of magnitude worse for RL, where unlike in SL [56], we have not witnessed results supporting a power-law scaling of test loss on new tasks, as a function of the amount of training data. Existing static RL simulators may thus impose a more severe data limitation than static datasets, which have been shown capable of inducing strong generalization performance [62].

2.3. Learning from unlimited data

These fundamental limitations of both SL and RL result from their inherently offline formulation, whereby the training data are collected once from a blackbox process and provided to the learning process *a priori*, in the form of a static dataset or predefined simulator. To keep its predictions relevant in an ever-changing world, a model must, instead, continually collect and train on new data—that is, it must perform continual exploration. We call such a data collection process *generalized exploration* when it seeks to explore the full space of possible input data to the model. When these data space is unbounded, we may say the process performs *open-ended exploration*, and a model trained on such a data stream then performs *open-ended learning*. Though RL considers the problem of exploration, existing methods insufficiently address this form of data collection.

To see why this is the case, we first note that the MDP (i.e. the simulator) plays an analogous role in RL to the training data in SL: Both define some distribution $\mathcal{D}$ over the space of available training data. In the case of standard SL, this distribution $\mathcal{D}$ consists of a uniform distribution over the individual data points $\{(x_i, y_i)\}_i$, where $(x_i, y_i) \sim P(X, Y)$. In RL, the simulator returns a training trajectory $\tau = (s_0, a_0, r_1, \ldots, s_T, a_T)$ for each policy π and starting state, s_0. The distribution $\mathcal{D}$ is then the marginal distribution $P(\tau)$ under the joint distribution $P(\tau, \pi, s_0)$ induced by the MDP, some distribution over policies π, and the initial state distribution $P(s_0)$. In practice, the initialization of the policy network and RL algorithm determine the evolution of π within the simulator, and thus both $P(\pi)$ and $P(\tau)$. We then see that exploration within the simulator is equivalent to sampling from $P(\tau | \pi, s_0)$ for a larger set of distinct π and s_0 values, thereby increasing the support of $P(\tau)$, and consequently the number of distinct trajectories seen during training. Such a process can be viewed as a form of *prioritized sampling*, in which data points favoured by the exploration process—perhaps due to some form of novelty or uncertainty metric—are sampled with higher frequency. Applying standard RL exploration methods to SL then corresponds to actively sampling from $\mathcal{D}$. Importantly, such exploration methods do not expand the data available for training, but only the order and frequency in which the model experiences the existing data already within the support of a predefined, static distribution $\mathcal{D}$. For example, we can use such exploration algorithms to encourage our virtual assistant to uncover diverse and reward-rich trajectories during training, but the resulting data remain confined to the set of experiences represented by the static simulator, just as its conversational training data is limited to the exchanges captured in a static corpus.

Addressing the data limitation of SL and RL requires a different formulation of exploration, one in which the exploration process continually seeks new data to expand the support of the training distribution $\mathcal{D}$, so to provide the model with useful information—that is, where the model may experience the most learning. Achieving generality then requires the model to continually retrain on such newly acquired data—effectively learning on an unlimited stream of data at the frontier of its capabilities. Importantly, for RL, this entails discovering or inventing whole new MDPs, in which to collect transitions offering high information gain, e.g. those leading to the highest epistemic uncertainty as estimated by a Bayesian model [63–66], regret relative to an oracle [67], gradient magnitude or gain in model complexity [68]. The question of what data should be collected for further training the model resembles that addressed by prioritized training approaches, including both active learning [69,70] and curriculum learning methods [68,71], but differs in a subtle and important manner worth reiterating: prioritized training seeks to select the most informative data points for learning (and for which to request a ground-truth label in the case of active learning). Crucially, prioritized training assumes that the training data are otherwise provided *a priori*. By contrast, we call our problem *active collection*, whereby we must gather the most informative data for further training in the first place, and thus it is one that must precede any form of prioritized training. In defining active collection, we explicitly make clear the separation between *optimization*, the process of fitting a model to data, and *exploration*, the process of collecting these data. As we have argued, by training on static data and simulators, SL and RL implicitly presuppose that the problem of active collection has been solved, turning a blind eye to this critical enabler of learning. In accelerating, the rate of information gain, active collection may be especially important for unlocking scaling laws for transfer performance to new tasks in RL, similar to those observed in SL.

In general, there are two ways to gather additional training data:

Offline collection occurs through a dedicated process separate from the model. For SL, this might entail using a web crawler to amass a comprehensive dataset of song lyrics, if it is found that the current assistant performs poorly in this domain. For RL, this might entail extending the simulator to include additional states, actions, transitions and rewards that model important dynamics deemed missing. This process is equivalent to that used to collect the initial training data, with the exception that subsequent data collections can condition on the previous training data and current model performance in order to maximize some notion of information gain. Being largely driven by human expertise and ad hoc intervention, offline collection can be costly and susceptible to human biases.

Online collection uses the model's own interactions with its deployment environment as additional training data. Any deployed model may be considered such an interactive ML system, which changes its environment with its own predictions [72]. In SL, this might entail simply retraining the model on a dataset of real-world interactions, e.g. whether the user engaged with a recommendation or found a response to a query useful. In a purely RL setting, this might entail collecting trajectories in the deployment domain that are used to fine-tune or extend the simulator. In this case, the simulator may be a world model [73–75]—a DNN parameterization of the MDP that approximates the transition, reward, and observation functions—which can be retrained on this additional data. However, retraining on the model's own interactions can reinforce the model's existing biases.

Importantly, the potential biases of data collection can be mitigated by ensuring the data are *actively* collected—that is, the collection process should seek datapoints maximizing some notion of potential information gain for the model. This collection can occur through a mix of manual guidance via a human expert and self-supervised approaches that directly generate or search for such highly informative datapoints. Active collection generalizes the form of exploration typically studied in RL, extending the domain of exploration beyond a single MDP and into that of an unbounded space of MDPs. By performing *active offline collection*, we directly train the model on data that can effectively strengthen its known weaknesses and reduce the chances of collecting redundant data. Similarly, by

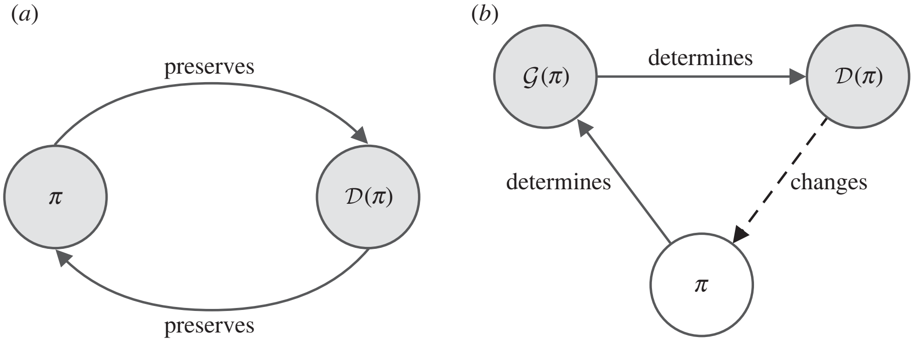

Figure 2. (*a*) The bootstrap problem, due to performative prediction, leads to an equilibrium whereby the model fully determines a stationary data distribution $\mathcal{D}(\pi)$ that does not change upon retraining. As a result, both π and $\mathcal{D}(\pi)$ become deadlocked in a fixed point. (*b*) Open-ended exploration delegates data collection to an active collection process $\mathcal{G}(\pi)$, which produces data $\mathcal{D}(\pi)$ designed to improve the current policy π by presenting it with data on which its performance is weak. Under this system, π continually improves, thereby avoiding fixed points.

performing *active online collection*, we selectively sample only the most informative interactions for learning, which tend to be those that challenge the model's existing biases. Crucially, as we discuss in §3, active collection avoids the highly problematic outcome of falling into local optima that can result from a ubiquitous bootstrap problem in online learning.

3. The bootstrap problem in online learning

Indeed, simply collecting data online in the deployment domain, without actively seeking it, is unlikely to produce an increasingly generally intelligent model. More likely, the model will fall into a local optimum: In online learning, the model retrains on newly acquired data, seeking to maximize an objective. Problematically, the data collected online is a function of the model's own predictions. For example, the state of the policy (e.g. the model parameters, if it is directly parametrized) within an RL agent influences the experiences it will gain from interacting with the environment, thereby influencing the data from which the agent itself will be updated. Left unaddressed, this causal loop between the data-generating process and the model's own predictions can result in a *bootstrap problem*, where this feedback loop further amplifies any systematic errors or inherent biases in the model's predictions.

Returning to our running example, we may decide to train our virtual assistant to shop for us on the web, where it consistently finds good deals on Amazon. As our assistant visits Amazon more often, it collects more online data of Amazon's webpages, which is used to refine the web simulation used for training. As the assistant becomes more adept at navigating Amazon, it consistently attains high rewards when fulfilling our requests there, further reinforcing this preference, which then grows to the extent that it never visits any other websites, where there may be much better deals. Iterated retraining on data collected online thus causes our agent to become stuck in a local optimum. This bootstrap problem, depicted in figure 2, impacts all online learning systems. It has been studied across various problem settings, including in contextual bandits, where it goes by the name of *one-sided learning* [76,77], and classification, under the game-theoretic lens of *strategic classification* [78].

This bootstrap problem has been formalized in the problem setting of *performative prediction* [79]. Here, the model f induces a distribution $\mathcal{D}(\theta)$ over the training data Z. The problem arises from the equilibria of this dynamic system, between model and data distribution. At such fixed points, the data distribution induced by the model, with parameters θ^*, is identical to that on which it was last retrained and which induced the solution θ^*. With these definitions, we can define the problematic points θ^* in the model's parameter space as those satisfying

$$\theta^* = \arg\min_{\theta} \ \mathbb{E}_{Z\sim\mathcal{D}(\theta^*)}[\mathcal{L}(Z, \theta)]. \tag{3.1}$$

While prior analysis focuses on the convergence toward such fixed points and on their optimality, these are exactly the points that agents seeking to learn general capabilities must avoid: at such fixed points, the online learning system stagnates into a subset of training data, preventing the possibility of learning anything beyond the current data distribution. Despite the bootstrap problem, online retraining is commonly implemented in production systems [80–84], while this crucial issue remains largely unaddressed, resulting in unfavourable emergent phenomena, such as extremization through personalized content recommendation systems and the amplification of socioeconomic disparities through AI-powered lending systems [85–88].

Standard exploration techniques in RL and prioritized training techniques in SL have been proposed to address this issue. Exploration and prioritized training both select the next training data points that are, by some criterion, those on which the model is most likely to be wrong, thereby forcing the model to reconsider its decisions and thus deviate from any premature equilibrium. In this sense, prioritized sampling could be described as a form of exploration, and conversely, exploration in RL, as a form of prioritized sampling. Such methods may suffice to avoid local optimum due to the bootstrap problem in a limited-data setting, though of course, reaching the globally optimal solution still ultimately corresponds to falling into a fixed point, beyond which no further learning occurs. The central, limiting factor is that existing exploration methods target a limited data setting, operating on either a fixed dataset or a static simulator. In order to go fully beyond exploratory equilibria, we must rethink exploration to encompass exploring beyond any one dataset or simulator.

4. Open-ended exploration

How can we perform exploration beyond a single static dataset or simulator? Before tackling this challenge, it is valuable to consider how exploration is currently performed in static, limited-data settings. Exploration in RL typically aims to maximize some criterion of novelty or uncertainty [89]. When the same evaluation criterion is used to select the next datapoints for training in SL, the process reduces to prioritized training. In this section, we centre our discussion on exploration in RL for a static simulator, then consider how such strategies can translate to supervised learning.

4.1. Exploration in a static environment

Let us call the RL policy that seeks to maximize the discounted return in a given MDP the *exploitation policy*. As discussed, the exploitation policy can become trapped in local optima, due to the bootstrap problem. Exploration seeks to address this issue by producing a second policy, which we call the *exploration policy*, to collect training data for the exploitation policy that is unlikely to be generated by simply following the exploitation policy. In order to find such data, the exploration policy typically either performs random actions in each state or seeks to maximize a measure of novelty—that is, transition data that looks different from what it has previously experienced, which may be computed on an episodic or lifelong basis. A common measure of novelty for state s is based methods for counting state visitations, whereby less visited states are consider more novel [90]. In practice, there are many proxy measures of novelty, like epistemic uncertainty, which may be estimated by the variance of a prediction network [91,92]. Similarly, prediction error [93,94] or regret—the difference between optimal return and that achieved by the agent—can also be used as measures of novelty, assuming the agent fares worse on environments that present novel challenges. Importantly, when applied to the mean performance over a batch of trajectories, these latter approaches implicitly subtract away sources of inherent, irreducible uncertainty [64,95] that can act as *stochastic traps* for a novelty-seeking agent. Alternatively, such irreducible uncertainty can be removed from the calculation by explicitly modelling it [96]. In general, the exploration policy aims to maximize an *intrinsic reward function*, $I : S \times A \rightarrow \mathbb{R}$ [97].

Beyond the shared motivation for seeking informative transitions, methods differ in how exploration is folded into training. One common tactic is to maintain a separate exploration policy, e.g. one that learns to maximize the future novelty of the exploitation policy. In this setting, the exploration policy is typically called the *behaviour policy*, as it is solely responsible for data collection, while the exploitation policy is called the *target policy*. The target policy then trains on the transitions collected under the behaviour policy using importance sampling [23,98]. Another increasingly popular approach is to use a single policy to serve as both exploration and exploitation policies [99–103]. This single policy takes actions to maximize a weighted sum of extrinsic and intrinsic returns. As exploration continues to reduce the uncertainty in most states of the MDP, the intrinsic reward tends to zero, resulting in an approximately purely exploiting policy at convergence, though in general, annealing the intrinsic term may be required [104]. A related set of approaches based on *probability matching* samples actions according to the probability that each action is optimal, where actions may receive some minimum amount of support to encourage exploration, or support that increases with the estimated uncertainty of the transition resulting from taking the action [105,106].

4.2. Exploration in environment space

Our discussion of exploration in a single, static MDP suggests close parallels to the more general problem of active collection, with the major difference being that exploration methods in RL focus on a single, predefined MDP—as defined by the static simulator—rather than over the infinite set of all potential MDPs, as sought by active collection. Exploration in the space of parametrized environments offers a path towards this more general form of exploration.[1]

4.2.1. Exploring a subspace of environments

In recent years, RL research has reoriented around learning optimal policies for distributions of environments [25,55,107]. In this expanded setting, the environment is parameterized by a set of *free parameters*, whose values alter properties such as the environment topology, entities present, transition dynamics, and the underlying reward function. The choice of free parameters specifies a particular subspace of environments, within the space of all possible environments. Simply randomizing over these parameters during training, in what is called *domain randomization*, can lead to robust policies. Automatic curriculum learning (ACL) methods [108], which produce adaptive curricula over these parameters have been shown to further improve robustness. Such adaptive curricula adjust the free parameters of the environment throughout training to maximize a measure of the agent's learning potential, resulting in an adaptive curriculum over specific environment configurations. Recently, unsupervised environment design (UED) [67] extends the problem setting of *decision-making under ignorance* [109] to capture the notion of robustness and frames the problem such that solutions naturally take the form of adaptive curricula. In UED, the MDP tuple is augmented with an additional set of free parameters, denoted Θ, whereby possible environment configurations correspond to specific settings θ of these parameters. An adaptive curriculum then arises from a game between a teacher, which proposes configurations θ, and a student that learns to solve them. The choices of payoffs for the student and teacher then correspond to an infinite spectrum of possible adaptive curricula. For example, DR corresponds to a constant payoff for the teacher—so that it is indifferent to the value of θ, and thus randomizes over it. When the game is zero-sum, with the teacher receiving the regret of the agent as its payoff, any Nash equilibrium [110] corresponds to the student playing a policy that minimizes the worst-case regret—that is, the minimax regret policy [111].

Curriculum games, in which the teacher seeks to generate environment configurations where it is possible for the agent to rapidly improve, lead to co-evolutionary dynamics that gradually increase the difficulty of these configurations, as the student grows more adept. Importantly, the teacher's payoff $C : \Theta \rightarrow \mathbb{R}$, should adapt to the agent's capabilities by satisfying these criteria for high learning potential:

(i) **Improvability:** The agent does not fully succeed. There is room for improvement.
(ii) **Learnability:** The agent can efficiently learn to improve.
(iii) **Consistency:** The solution to each environment configuration is consistent with those of other configurations.

If the proposed environment configurations do not satisfy (i), then the configuration provides no additional benefit for training, as the current model already performs optimally. If the configuration does not satisfy (ii), then despite there being room for improvement, the current model is unlikely to improve when training on that configuration. Presenting many configurations that fail to meet the learnability criterion can stall learning progress, as the agent struggles to learn the solution. Finally, (iii) ensures that optimal behaviour in a proposed configuration does not conflict with that in another, e.g. entities are semantically consistent. As in exploration in a singleton MDP, learning potential estimates must remove sources of aleatoric uncertainty to avoid stochastic traps. Such corrections can be done by directly modelling and subtracting sources of aleatoric uncertainty or estimating mean learning potential over a batch of trajectories for each configuration.

[1] The set of all MDPs can in general be indexed into a single *universal MDP* by assigning each a unique real-value index and including this index in the state. Exploration over the space of MDPs can then equivalently be viewed as exploring this universal MDP. We instead frame the discussion in terms of the space of MDPs to emphasize and exploit the inherent modularity of the problem.

Just as exploration methods in a singleton MDP encourage the agent to visit novel parts of the state space to maximize opportunities for learning, adaptive curricula bring the agent to the environment configurations that offer the most learning potential, itself a form of novelty. In this way, adaptive curricula perform exploration in a parameterized space of environments.

4.2.2. Exploring the full space of environments

Still, a parametrized environment only allows agents to learn to succeed in the specific configurations expressible within its parametrization. Since the parameters only modify the instantiation of a static simulator designed to model a limited task domain, the agent's learning will eventually plateau. Open-ended exploration requires an open-ended simulation, one that continually expands to encompass new domains—in essence, exploring the space of possible MDPs. To this end, we can define a search process $\mathcal{G}(\pi)$ that takes the current policy π and returns a distribution $\mathcal{D}_{\mathcal{M}}$ over the set of MDPs, i.e. a countably infinite set of programs under some Turing-complete language. This distribution aims to focus its support over MDPs that maximize the learning potential criterion, $C:\mathcal{M}\times\Pi\rightarrow\mathbb{R}$. By evaluating π on $m\sim\mathcal{G}(\pi)$, we can iteratively update $\mathcal{G}(\pi)$ toward distributions over $\mathcal{M}$ that place greater weight on MDPs with higher learning potential. In practice, $\mathcal{G}$ can consist of large parametric models alongside non-parametric models and human-in-the-loop components to assist with the search process. Moreover, the exact components implementing $\mathcal{G}$ may shift over time. The search process $\mathcal{G}$ then seeks MDPs $m^*\in\mathcal{M}$ maximizing the exploration criterion:

$$m^* = \underset{m\in\mathcal{M}}{\arg\max}\; C(m,\pi). \tag{4.1}$$

In practice, we might model $\mathcal{G}$ as a large generative model, whose outputs are MDP programs, and its weights, fine-tuned to maximize the exploration criterion, C. Left alone, this approach runs into three interrelated issues: first, the programs generated will be largely malformed, and an astronomical number of iterations may be necessary to discover the first valid MDP program. Second, the programs discovered will likely be nonsensical, failing to reflect the target problems of interest. Third, upon discovering valid programs, exploration may overly focus on a limited subset of the full space of programs. In order to generate both well-formed and relevant MDPs, we must constrain the search to begin near well-formed programs reflecting the target tasks of interest. We use the phrase *grounding with respect to* $\overline{\mathcal{M}}$ to refer to the process of enforcing this constraint with respect to a target set of seed MDPs, $\overline{\mathcal{M}}=\{m_k\}_{k=1}^{K}$. In order to maintain diversity in the programs explored, we can directly encourage diversity in the space of MDPs. Given distance functions Δ_D and Δ_G between two programs, we can ground open-ended exploration with respect to $\overline{\mathcal{M}}$ and encourage diversity by augmenting the exploration criterion in equation (4.1):

$$m^* = \underset{m\in\mathcal{M}}{\arg\max}\left(\underbrace{C(m,\pi)}_{\text{learning potential}} + \underbrace{\alpha_D\sum_{m_i\in\hat{\mathcal{M}}}\Delta_D(m,m_i)}_{\text{diversity}} - \underbrace{\sum_{m_k\in\overline{\mathcal{M}}}\alpha_k\cdot\Delta_G(m,m_k)}_{\text{grounding}}\right), \tag{4.2}$$

where α_D and α_k are positive weights modulating the relative importance of the diversity bonus and being close to $m_k\in\overline{\mathcal{M}}$, respectively, and $\hat{\mathcal{M}}$ is finite queue of the top solutions found so far. We explicitly define two distinct distance functions as, in general, the diversity and grounding terms might consider different notions of distance, e.g. the string edit distance or a distance in the latent space representation of a pretrained generative model of programs. When computed with respect to specific attributes of generated programs, Δ_G may encode inductive biases of what useful MDPs look like, e.g. whether a given MDP simulating a physical system produces trajectories consistent with the laws of physics. When humans are part of the open-ended exploration process, they can regularly modify the set of seed MDPs, $\overline{\mathcal{M}}$, used for grounding, and steer exploration toward specific regions in the space of MDPs deemed important by adjusting the corresponding α_k weights. Figure 3 provides a toy illustration of how these criteria impact the search process.

It is important to draw a similarity between this exploration criterion and the objective of algorithms for quality-diversity (QD) [112], commonly explored in the evolutionary computing community. Such algorithms seek to find a diverse set of solutions, where the best candidates within distinct subspaces of the solution space are gradually discovered. Viewing the learning potential term as a measure of quality, we see that equation (4.2) implements a form of QD. However, this process remains

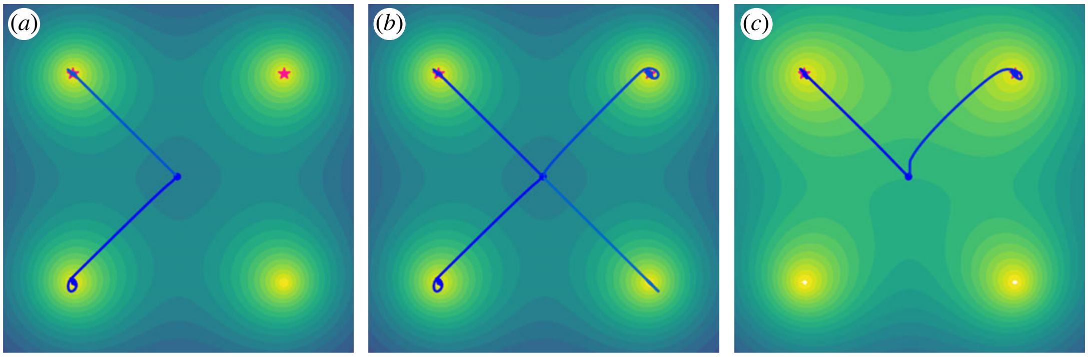

Figure 3. A simple visualization of how an open-ended exploration process searches for points in task space according to equation (4.1). Brighter regions have greater learning potential. The pink stars are tasks of interest, and the blue lines are search paths. (*a*) Searching only based on learning potential can lead to both useful tasks and irrelevant tasks. (*b*) Searching with an additional diversity term can lead to finding both more relevant and irrelevant tasks. (*c*) Grounding the search to useful tasks leads to finding more relevant tasks.

significantly distinct from QD, as the measure of learning potential is a function of the current model, and therefore non-stationary throughout training. Moreover, learning potential is not transferable as a quality measure across agents in different training runs. To acknowledge the similarities to QD while highlighting its distinctness from QD, we call the problem of finding diverse data with high-learning potential that of *learning-diversity*.

Similar to the outlined open-ended exploration process, prior work on the POWERPLAY algorithm has considered finding increasingly general agents by jointly searching through the space of MDP programs and that of solution programs [33]. We detail the relationship between our proposed approach to open-ended learning and POWERPLAY in §6. Importantly, POWERPLAY frames the problem purely as a form of iterated optimization via random search, ultimately a computationally infeasible approach. By contrast, equation (4.2) defines an exploration criterion allowing for agents to gradually increase their generality by actively collecting and training on data from diverse regions of the MDP space with high learning potential, while remaining grounded and steerable by expert guidance, e.g. by a human. Importantly, this generalized form of exploration not only applies to RL in MDPs, but can also be straightforwardly translated to SL.

4.3. Exploration in data space

In viewing a static SL dataset as a parametrized, single-step MDP, we can similarly implement open-ended exploration by pursuing the criterion in equation (4.2). For a static dataset $\mathcal{D} = \{x_i, y_i\}_{i=1}^{N}$, where $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$, we can reframe the problem as finding the optimal policy for an MDP m, whose optimal policy f is the model that minimizes the empirical risk on $\mathcal{D}$. In this case, $m = (S, A, T, R, \gamma, p)$, where the set of states is defined as $S = \{x_i\}_{i=1}^{N}$, the set of actions $A = \mathcal{Y}$, the transition function T simply terminates the episode, and the reward function R returns the negated supervised loss between the predicted value $f(x)$ and target value y. Since the MDP is single-step, γ can be any value in [0, 1]. The initial state distribution p is a uniform distribution over S. As defined, the optimal policy f for m must minimize $\mathbb{E}_{x_i \sim p}[\mathcal{L}(f(x_i), y_i)] = (1/N)\sum_{i=1}^{N} \mathcal{L}(f(x_i), y_i)$, exactly the empirical risk over $\mathcal{D}$, as typically minimized under SL.

Having established the correspondence between SL datasets and single-step MDPs, we see that open-ended exploration in SL entails active collection of datasets that maximize the exploration criterion outlined in equation (4.2). This view makes clear that sampling strategies used in prioritized training implement specific instantiations of this criterion, limited to a fixed training dataset, $\mathcal{D}_{\text{train}}$. For instance, active learning selection functions, as well as those in curriculum learning, are largely based on choosing the next training sample to maximize the information-gain, or relatedly to reduce some notion of uncertainty over the model's own predictions or variance in performance error [69,113]. Any such selection function corresponds to a specific choice of the learning potential subcriterion, $C(m, \pi)$, where m now corresponds to a datapoint in $\mathcal{D}_{\text{train}}$, and π, the supervised learning model. Curriculum-based prioritized training largely mimics the same selection strategies [68,71,114], while removing the interactive label queries of active learning.

This framing also points to the rich possibilities that lie in extending such prioritized training methods toward the full scope of open-ended exploration: New variants of prioritized training may benefit from additionally incorporating approximations of the diversity and grounding subcriteria. The diversity term helps prevent oversampling of particularly challenging points at the expense of other informative points. The grounding term comes into play when incorporating various kinds of generative procedures based on the training data to perform data augmentation, ensuring generated samples do not go too far afield from the real data. Early results suggest such generative models hold great promise for improving the robustness of SL models when used to generate *synthetic training data* [115,116]. Such data generation may be particularly effective for amplifying the amount of available data for rare samples, e.g. extreme weather conditions or unique vehicular appearances and interactions captured by camera feeds for training the models underlying a self-driving car.

Importantly, by generating *synthetic data* grounded to real task domains, the exploration process can produce alternate views of existing data that serve as interventions of specific causal features within the data space [117–119], which can lead to models whose predictions are more consistent with the underlying causal structure of the task domain [48]. As data generators, such generative models play an analogous role to the simulator or world model in RL, providing alternative views, and allowing the model—guided by an appropriate exploration critrion—to probe the data space through trial and error or more sophisticated experimentative behaviours. As in the case of RL, online data collected in deployment can be used to further fine-tune the generative models to continually expand the space of represented data. More generally, as in RL, we can pursue the open-ended exploration criterion in SL through a combination of curating useful interaction data (i.e. online collection), manually collected seed datasets that reflect our domains of interest (i.e. offline collection), and actively generating data with high learning potential for our model using powerful generative models such as state-of-the-art diffusion models [120–123], grounded to these seed datasets.

5. A unified view of exploration

Our discussion reveals that exploration can be reconceptualized as a search process over the space of MDP programs. Such a view of exploration highlights direct correspondences between prioritized training methods in SL and exploration in RL, while radically expanding the domain of exploration to include the full space of possible training data. Crucially, exploration remains the same process of finding the most informative datapoints for learning, regardless of the the optimization procedure used—whether based on an SL or RL objective. Open-ended exploration performs this search over the full data space and serves as the core driver of information gain in open-ended learning systems seeking general capabilities. We now ground these ideas to a concrete example, illustrating how such open-ended exploration might be implemented in practice to give rise to a learning system of increasingly general intelligence. The differences between our hypothetical open-ended learner and ad hoc forms of exploration used in practice today then reveal a new data-centric research frontier.

5.1. A unified example

Let us think through how a concrete system might perform such open-ended learning by returning to our example of a virtual assistant. Recall that our virtual assistant must perform two key functions: (1) converse with a user via text, and (2) navigate any website to accomplish tasks on behalf of the user. Either (1) or (2) in itself already approaches the difficulties of training a generally intelligent agent; both language and software offer universal representations of any computable task, so becoming generally competent in either domain entails achieving a form of general intelligence—the ability to successfully handle any computable task. Of course, the space of all computable tasks (i.e. $\mathcal{M}$) is vast and even humans are not generally intelligent in this regard. Nevertheless, we can pursue continual incremental progress toward this goal by training an IGI through open-ended learning. Such a system, summarized in figure 4, performs three distinct, interlocked modes of learning.

Initial training. As usual, we design an initial dataset to seed the training of our virtual assistant. As done in the training regimes of current state-of-the-art conversational models, this dataset likely includes a large corpus of dialogue mined from conversations between real people. As performed in recent works, RL can be used in combination with a human-designed reward function, i.e. reinforcement learning with human feedback [124,125], to encourage responses that are both consistently friendly and helpful. Following the example of other grounded interaction models like LaMDA [126], web navigation

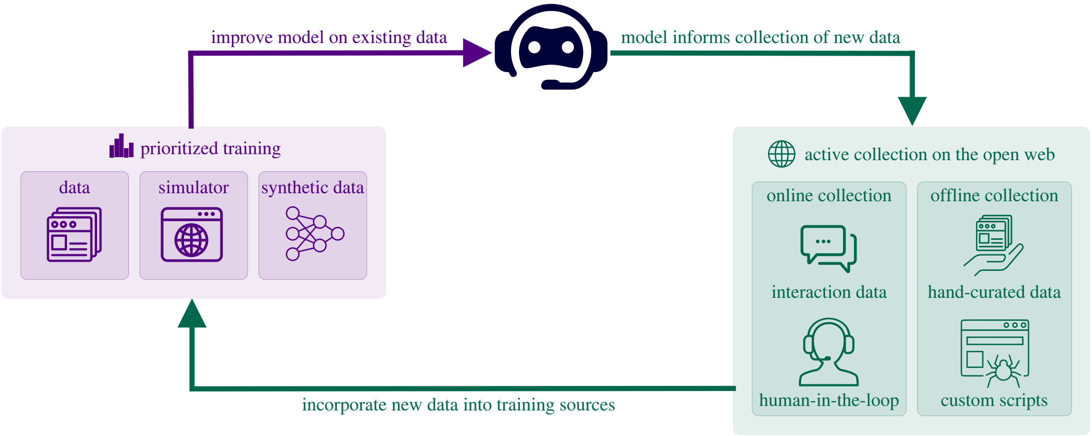

Figure 4. An overview of how active collection and prioritized training combine to enable open-ended exploration and learning to produce a virtual assistant with increasingly general capabilities. An initially trained assistant model is used to inform the active collection of new data from both offline and offline sources, possibly with assistance from humans and other programs in the loop. The newly collected data is then used to update the previous training sources, which might include labelled datasets, simulators, and synthetic data, e.g. sampled from generative models or a learned world model.

agents can similarly be bootstrapped from a combination of SL over a dataset, perhaps collected from trajectories generated by hand-coded controllers for performing common tasks on a diverse set of webpages, and RL inside of a web navigation simulator [127–129]. To connect its conversational and web navigation functions, synthetic and augmented conversation data, both generated offline and within a simulator, can serve as templates for successful dialogues whose responses require web navigation.

Active collection. After initial training, we can deploy our assistant to users. At deployment, we perform active online collection, curating a dataset of interactions in which the assistant performed poorly, according to some proxy metric such as whether users engaged with the content shared by the assistant. In parallel, active offline collection can be performed, whereby specific weaknesses in the model are identified, e.g. websites where navigation commonly fails or semantic clusters of request types that result in low engagement. These areas are then targeted for the collection of real dialogue data and expert web-navigation trajectories, resulting in an *adversarial* dataset. This collection can occur through many channels, e.g. manually through crowdsourced efforts; storing the interactions resulting from a human-in-the-loop taking over for the assistant when the assistant's predictions become highly uncertain; and custom webcrawlers that harvest relevant conversational data or videos from YouTube that serve as expert demonstrations [130]. The new states, actions and dynamics present in this data can further be integrated into the existing simulator, either through fine-tuning if the simulator is a learnable model, e.g. a world model, or through manual coding. Further, these data can be used to train a generative model, grounded to the real data, thereby greatly amplifying the amount of data and allowing for fine-grained exploration of the relevant regions of the data space. The assistant can then be retrained on this adversarial data to become more robust. This active collection and retraining cycle then repeats, resulting in an agent that becomes robustly capable across an increasing number of domains. Importantly, we see that exploration naturally occurs over the data space itself—not within some notion of a static simulator or dataset. This example also highlights the blurring of the lines between simulated and real data. Real data are used to generate further simulated data, and performance on simulated data informs the active collection of real data.

Prioritized training. The assistant performs prioritized training to efficiently learn from the collected data. In SL, there exist many prioritized training methods [68,131] that can be used to selectively sample datapoints that offer the most learning potential. In RL, methods like Prioritized Experience Replay [71] serve the same purpose. Other, more recent *unsupervised environment design* [67,128,132,133] approaches mix active collection and prioritized training, whereby the algorithm actively searches for new MDPs, e.g. new website configurations, that offer the most promising environment configurations for learning, and then actively samples these configurations during training. Viewing a generative model as a parametrized dataset, we can apply the same approaches to both explore the data space of the generative model and actively sample the discovered points for training.

5.2. A view beyond the frontier

Many real-world systems, including those controlling large-scale applications such as video recommendation [134] and traffic prediction [135], use ad hoc versions of iterated retraining, relying on online and offline collection processes that are largely agnostic to learning potential. As such, their online collection can suffer from the bootstrap problem, falling into premature equilibria. Similarly, their offline collection may lack sufficient exploration of the data space. While we do not offer a specific solution for effectively pursuing the open-ended exploration criterion in equation (4.2), we believe articulating these criteria can be useful in motivating future work in this area.

As our state-of-the-art learning methods approach asymptotically-optimal performance for static training sets and benchmarks, the primary lever for improving our models lies in improving the training data itself. Several results support this data-centric view of ML: for large neural models, continually scaling up the quantity of training data empirically improves generalization performance [17,56,136]. Generating additional supervised samples through methods like *data programming*—which generates new labels based on human-provided, approximate labelling functions, results in significant performance gains in several domains [137,138]. We foresee the pendulum of research will swing from *learning innovation* in the form of better model architectures and optimizers for training on specific data to *data innovation* in the form of more intelligent algorithms for collecting additional data for improving a specific model. This turn toward data innovation is the natural conclusion of the paradigm shift in software introduced by the advent of deep learning. This shift transitioned software development from the largely imperative approach of directly designing solutions in code to the declarative approach of simply designing an objective function over training data that is optimized by a DNN—an approach sometimes referred to as *Software 2.0* [139,140]. The solution then emerges as a result of this optimization, but the choice of training data remains a highly manual design problem. The data innovation paradigm then goes one step further, by also replacing the direct design of training data with the design of the exploration criterion, that is, a self-adapting data-generation process. We may thus refer to this data-centric paradigm as a kind of *Software Squared*. Like the shift from designing solutions to designing objectives, we believe the shift to designing the data-generation process itself will change the face of computing and enable a new generation of more generally intelligent software.

6. Intelligence as capacity for exploration

Open-ended exploration presents a promising path to the longstanding goal in AI research of developing increasingly more powerful Artificial General Intelligence (AGI)—that is, systems capable of matching or exceeding humans in an ever wider range of tasks. The very act of defining a general notion of intelligence in humans and animals has been historically fraught with debate. Unsurprisingly, the quest to produce an agent implementing general intelligence is no less contentious. Many alternatives toward this grand goal have been proposed. Here, we relate the open-ended path to general intelligence to the most relevant alternative paths.

The precise definition of AGI is ultimately a subjective one, based on the specific criteria and assumptions held. As previously mentioned, we take the view that general intelligence cannot be quantified in degree, as such a computation is necessarily intractable given the countably infinite size of the space of possible, computable tasks. Rather, we can only compare the relative degree of generality held by two models A and B, with respect to a specific set of tasks, $\mathcal{T}$. We can then say A is more general than B in $\mathcal{T}$, if A performs above a performance threshold on more tasks in $\mathcal{T}$ than B, such that A also at least matches the performance of B on all tasks that B performs above the threshold.[2] Under this definition, a learning process may hold the capacity to become increasingly general over time, by incrementally mastering a wider range of challenges. Exploration, as we have argued, serves as the central mechanism enabling this gradual expansion of capabilities.

AIXI [11], perhaps the most prominent model of AGI, runs counter to our viewpoint. AIXI is the globally optimal solution to the Universal Measure of Intelligence, Υ [142], defined in (6.1).

$$\Upsilon := \sum_{\mu \in \mathcal{M}} 2^{-K(u)} V_{\mu}^{\pi} \tag{6.1}$$

[2] Naturally, this definition implies that there may be no single most general agent in some domains, such as zero-sum games, for which the ordering of strategies is not guaranteed to be transitive [141].

This measure, Υ, defines the degree of general intelligence for a decision-making policy π as the sum of the expected performance of π in every computable MDP $\mu \in \mathcal{M}$, as captured by its value function, V^{π}_{μ}. Crucially, the contribution of each MDP μ is inversely weighted according to its Kolmogorov complexity, $K(\mu)$, the length of the shortest program, represented as a bitstring, that implements μ. A shortcoming of this definition is that Kolmogorov complexity is generally non-computable, making Υ, and thus AIXI, non-computable with any finite compute budget. While there exist computable approximations of AIXI for a limited program length or finite compute budget, such solutions remains computationally infeasible beyond the most toy MDPs [11,143]. Further, the assumption of a universal prior over all MDPs serves as a rigid, subjective decision that limits the true generality of this definition. Ultimately, such *top-down* definitions of general intelligence primarily offer insightful, theoretical sounding boards, but in practice, suggest no tractable path to what they describe.

By contrast, our *bottom-up*, exploration-driven view of general intelligence traces back to earlier works on self-improving algorithms, which propose open-ended search mechanisms for finding guaranteed improvements to a problem-solving program. Gödel machines continually search the space of solver programs and attempt to prove whether each candidate is more general than the current solver, which is then replaced by the better candidate [10]. However, this method entails the exorbitant computational costs of brute-force proof search. Its successor, POWERPLAY instead seeks to find the simplest modifications to a program that results in a more general solver [33], though this remains a costly search problem. Here, a simplicity constraint is enforced by limiting the compute budget available for searching for a new, yet unsolvable task and its corresponding modification to the solver. Similarly, prior work on Never-Ending Learning [144] frames an open-ended learning process over an infinitely expanding knowledge graph, though its task space is limited to classification and inference tasks on this growing ontology. Our framework of open-ended exploration provides a generalization of these approaches to data-driven agents operating over the full space of possible tasks. Importantly, active collection of new training data eases the costlier parts of search over the task space by explicitly considering the current capabilities and learning dynamics of the current model. By adapting the active collection criterion's relative weighing of learning potential, diversity, and grounding (as in equation (4.2)), open-ended exploration mitigates the potential risk of the search process tunnelling into limited portions of the task space, which may be completely divorced from the problems of interest to the system designer. Crucially, because equation (4.2) rates novelty based on the agent's current predictive capacity and experiential history, exploration and the agent's capabilities interact in a mutually compounding manner: As the agent learns to achieve greater reward in more environments, maintaining exploratory novelty forces a gradual ratcheting of its mastery over an increasingly diverse set of environments. Conceivably, an agent that continues to explore and learn can eventually perform the set of all learnable skills within the domain of exploration.

A related vision for producing general intelligence stems from the open-ended evolution community within the field of artificial life (ALife), which is concerned with developing programs that replicate the emergent complexity characteristic of living systems [145,146]. Kickstarting a process that exhibits open-endedness—that is, the endless generation of novel complexity—is seen as a key requirement for achieving this goal. Such an open-ended process may then become an AI generating algorithm (AI-GA) [34], by producing an ecosystem of increasingly complex problems and agents co-evolved to solve them. However, such a system may constitute a large population of agents specialized to specific challenges. Moreover, exactly how such a co-evolving system might be implemented in practice remains an open question. We propose open-ended learning as a path toward not only an open-ended process, but one resulting in a single, generalist agent capable of dominating (or matching) any other agent in relative general intelligence over time. In this way, open-ended learning bridges the search for open-ended emergent complexity in ALife with the quest for general intelligence in AI.

7. Open-ended problems

Open-ended learning requires SL datasets and RL environments that act as open-ended generators of tasks and training experiences. To bring this discussion to a close and inspire future research, we describe a series of major open problems on the path to achieving such a system.

7.1. Open-world benchmarks

The nascent subfield of UED in RL has begun to consider exploration over the entire space of environments [67,132,147], and therefore over the full space of possible training data. Yet, these preliminary works still focus exclusively on static simulators that, despite being parameterized over a space of possible configurations, lack the ability to grow their potential configurations in open-ended ways, making them inappropriate for training an IGI. Similarly, popular benchmarks in SL, like GLUE [148,149] and ImageNet [150] typically centre on a static dataset. The more recent StreamingQA benchmark studies the ability of a language model to adapt to a changing data distribution [151], but ultimately also relies on a static dataset. Other recent benchmarks more closely match the signature challenges of open-ended learning—a continually expanding problem space and active collection of training data: BIG-bench invites researchers to collaboratively grow a repository of benchmark tasks for large language models [152]. BIG-bench can be seen as a slow, human-driven emulation of open-ended exploration of the task space. However, it does not truly capture the notion of open-ended exploration, as it considers only a finite set of human-designed tasks during training. Similarly, Dynabench seeks online collection of adversarial examples via humans and models in the loop, allowing for open-ended expansion of the training data within the relevant task domain [153]. However, Dynabench is currently limited to a pre-specified task. A truly open-ended benchmark must test the ability of a learning process to continually discover and solve new problems. Building such a benchmark requires finding solutions to several important, interrelated questions.

Q1. In which domain should we study open-ended learning?

A primary challenge in creating an open-ended benchmark lies in defining the problem domain. ML has historically focused on problem domains that afford simple definitions, via either a hand-coded simulator or a representative, curated dataset. However, open-ended exploration requires a problem domain that both (i) continues to grow with the learner (i.e. offers learning potential and diversity) and (ii) captures the quality of real-world tasks of interest (i.e. grounding). For this reason, current proposals for open-ended environments based on artificial open-world games such as MineCraft [154–157] and NetHack [25,158], while perhaps satisfactory for the ALife model of open-endedness based on unbounded novelty, are insufficient for the open-ended learning of a general intelligence. For example, a generally capable MineCraft agent cannot be trusted to perform any task beyond the confines of the game, like driving your car. This distinction is important: neither open-endedness nor open-ended learning in itself implies a process that achieves general intelligence. It is crucial to ground open-ended exploration to real-world tasks of interest, as emphasized in equation (4.2), and current open-ended simulators satisfy this requirement. One viable candidate is the Internet—itself, an open-ended system capturing the world's knowledge. Like our virtual assistant, an agent that freely explores the rich task space over the comprehensive ontology of the Internet would develop and understanding of our world and learn to take actions, mediated by software, that affect specific changes within it. Another possibility is to develop an embodied intelligence that can freely explore the real world, though this approach will likely need to be supplemented by learning in simulation. Further, it remains an open question as to what the right subset of the problem domain—whether Internet or real world—provides the most effective starting tasks from which an open-ended learner can progress to increasingly general capabilities. For example, should the learner begin by seeking to understand world dynamics by exploring video-centric tasks or first develop an understanding of language through conversational or information retrieval challenges?

Q2. How do we design scalable open-ended data generators?

An equally important and complex challenge is that of designing a computational system capable of representing an endlessly growing set of tasks. While any task may be defined as a program implementing a decision process, naively storing and searching all such programs discovered in a non-parameteric fashion, as proposed in prior works [33], is computationally infeasible. Open-ended learning requires a generative process capable of continually inventing new tasks, while storing only the most useful task designs in a compressed representation. Ideally, the number of parameters in such a generator grows much more slowly than the number of tasks represented. While we might imagine a large generative model, such as a code-generation model or world model, as playing the role of such a generator, it is unknown whether current model architectures and optimization methods are suitable for this kind of continual invention and compression.

Q3. How should an agent interface with an open-ended task space?

An open-ended learner must process inputs from an increasingly diverse observation space and make decisions over an increasingly large action space. The agent thus requires a generic interface between agent and environment, capable of adapting the input and output representations of the agent model to the task at hand. This interface may take the form of tools invented by the agent, e.g. real or simulated hardware, or even a program. Such tools may even be passed on to other agents, which may further evolve the tool for new purposes, leading to new evolutionary dynamics independent of the original inventor. Recent work proposes that tool invention be critical to the emergence of open-endedness [159]. The interface question is deeply related to the choice of domain for open-ended learning. In a virtual domain like the web, the interface may abstract and programmatic in nature, while in a physical domain, the interface may consist of sensors, actuators and physical tool use. In each setting, we expect an open-ended learner to progressively innovate its own interface, a process reflecting the arc of human technology.

Q4. How do we measure the extent of open-ended learning?

Even with the above problems addressed, there are no commonly accepted measures for tracking the degree of open-ended learning achieved—that is, some measure of increasing capability. Previously proposed measures of open-endedness cannot be adapted for this purpose, as they focus on measuring novelty [160–162], rather than model capability. In general, such novelty and model capability are unrelated. For example, a process that evolves an agent across an endless range of mazes, while progressively growing the size of the agent's memory buffer, may score highly in novelty, but remains limited in capability. Performance-centric measures based on the number of tasks on which the agent experienced improved performance, such as the ANNECS metric [163], suffer the same shortcoming: the learning process that fixates on the maze domain may see the agent struggle with new maze variations before solving them, thus propping up such measures without increasing general capabilities. One feasible option may be to simply track the diversity of tasks based on domain-specific criteria, but such a solution does not apply to all domains of interest. A general metric for open-ended learning would need to be domain-agnostic. Such a metric might consider both the agent's behaviour in discovered tasks and task novelty based on a general task representation.

7.2. Active collection

Given a benchmark accommodating open-ended learning, there remain several major open problems centred on active collection—the means by which the learning process selects and gathers new training data. Existing UED methods perform a limited form of active collection by generating new simulator configurations, though limited to those represented within a fixed parameterization. Similarly, recent methods in SL perform active collection by actively generating new data based on data augmentations of existing data [164]. Open-ended exploration requires rethinking such approaches to search beyond a single, fixed parameterization of the data-generation process.

Q5. How do we determine what data to acquire next via active collection?

Open-ended exploration requires searching for new training data across the entire data space. As argued in our construction of equation (4.2), this search should be guided by *a priori* criteria, such as the informational worth, i.e. learning potential, of data and its alignment with specific tasks of interest. As the learning potential and diversity terms in the open-ended exploration criterion in equation (4.2) are dependent on the current model, they present challenging non-stationary search objectives for active collection. Efficient search may require a compact latent representation of the data space, as well as the use of surrogate models to cheaply approximate the value of a datapoint under the search criterion. This latent space might correspond to the input context to a Transformer-based generative model of the data space or of data-generating programs. Latent-space optimization [165–168] and quality-diversity search [169–171] methods make use of learned low-dimensional representations and surrogate models in this way, but assume stationary objectives. Moreover, it is currently unclear how similar criteria can be used to plan manual data collection procedures, such as hand-coded data-crawling scripts or test-time hand-offs to humans-in-the-loop. Further, we lack principled ways to predict the relative efficacy of training on data actively collected manually, online or offline. Lastly, it is unclear how the relative weighing between the terms in equation (4.2) should evolve over time.

Q6. How can we scalably augment our training data?

Efficient simulators enable RL agents to train on billions of data points in the span of days [60,172]. A promising direction to achieve similar computational advantages in SL, where we lack a simulator, is to directly learn a simulator of the training data. Such a simulator might consist of a generative model [173–176] or a partially induced program that models the data-generating process, e.g. a grammar modelled after sequential data [165,177–179]. Such simulators can generate massive amounts of *synthetic data*, serving as promising, parametrized regions of the data space over which to perform active collection, as well as for performing data augmentation of existing training data.

Q7. How much prior knowledge should be used to ground exploration?

The grounding term in the exploration criterion focuses active collection on parts of the data space resembling specific tasks of interest. Such similarity constraints allow us to embed key inductive biases into the search process, e.g. ensuring discovered simulator settings are consistent with physical laws. However, a strong grounding term can also force the search process to ignore important regions of the data space that may provide higher learning potential for the model, despite lacking a close resemblance to our exemplar data. In the evolutionary computing community, it is well known that novelty search can often lead to simpler, more effective solutions than objective-based optimization, through the discovery of useful stepping stones that would otherwise be overlooked [35].

Q8. How do we safely perform active collection?

In prioritizing specific data points for training, active collection and prioritized training inherently distort the training distribution with respect to the ground-truth distribution in the real world. Such distribution shifts can result in biased models that induce harmful downstream effects when deployed. Recent works characterize the bias that can result from prioritized training in both SL and RL, alongside methods for correcting for this bias [180,181]. However, we lack principled means to understand how such biases may materialize in the open-ended learning setting, where data for tasks may be simultaneously, actively collected and thereby result in cross-task interference effects.

The online setting then introduces many more risks, as it necessarily entails deploying an IGI in the wild. These risks stem primarily from the unpredictability of open-ended learning processes. For example, in order to maximize the diversity of its experiences, a novelty-seeking IGI may perform acts that are harmful to humans interacting with it, e.g. showing a human user inappropriate content or misleading information may elicit novel user feedback data to the detriment of the user's psychology and perception of reality. Malicious actors may even attempt to steer an IGI's learning process to reinforce certain harmful behavioural patterns, e.g. the spread of misinformation benefiting the malicious party. At scale, such emergent behaviours and biases may lead to cascading effects that jeapardize the stability of society. Amodei *et al.* [182] provide an overview of open problems around AI safety, and Ecoffet *et al.* [183] provide a detailed discussion of how safety issues emerge in systems optimizing open-ended objectives.

7.3. Orthogonal challenges

We view these directions as the fundamental challenges of exploration. Other open problems, focused on optimization, directly impact an agent's capacity to explore, but are not core to the generalized exploration problems we consider. These orthogonal challenges, including catastrophic forgetting [184–186] and the challenges of gradient-based optimization of non-convex objectives [187–189] and differentiable games [190–192], have historically been major areas of focus for the ML community. An agent's capacity to explore hinges on its ability to recognize novel data, which assumes a mastery of past experiences. More efficient optimization will thus generally benefit exploration. Similarly, finding new challenges entails retaining solutions to those already mastered. Exploration thus stands to directly benefit from methods addressing catastrophic forgetting.

8. Conclusion

The ML research community has made incredible advances over the past decade. However, outside of limited pockets of research, these advances have been developed primarily under the assumption of a static dataset, or otherwise, algorithmically generated data of limited description length, largely beyond the control of the learning agent. Thus, the overarching perspective of the ML community has largely ignored the generalized problem of exploration in learning: The learning agent must perform active collection, that is, it must both determine and acquire the data most informative for expanding its present capabilities on a continual basis, by considering three key aspects of the data: learning potential, diversity, and grounding to real tasks of value. We have argued that formulating this notion for existing ML methods requires rethinking the prevailing paradigm of exploration developed in RL, expanding its scope from a single, static environment simulator to the full data space containing information relevant to an unbounded set of possible tasks. Such an open-ended exploration process serves as the data-centric driver of open-ended learning. We believe open-ended exploration provides a more viable, bottom-up path toward general intelligence than alternative top-down approaches proposed in the past. On a short time horizon, independent of AGI, most ML systems deployed in an open world setting, with real users and peers, must perform some notion of open-ended exploration and learning, or risk fading into irrelevance. On a longer time horizon, the arc of such real-world ML systems is toward increasingly general capabilities, as continued improvement for most open-world tasks, from question-answering to driving, requires ever greater levels of sophistication in reasoning. Of course, our proposed, data-driven path to general intelligence is not without deep, unresolved open questions which we also discussed at length.

We emphasize that the goal of this discussion is not to determine an exact solution to the problem of open-ended exploration. Rather, we aim to sketch the outlines of the problem and its subcomponents, so that the community can begin to fill in the missing pieces under a common conceptualization. As our learning algorithms become ever more proficient at modelling any dataset, we stand to gain the most from thinking more deeply about what data we feed these algorithms. Open-ended exploration, in its essence, is the principled and continual collection of training data to maximize a model's performance across the data space. By directing the research community's own exploration to this much uncharted frontier, we are confident in making progress toward ever more general AI systems.

Data accessibility. This article has no additional data.

Authors' contributions. M.J.: conceptualization, investigation, methodology, visualization, writing—original draft, writing—review and editing; T.R.: conceptualization, writing—review and editing; E.G.: conceptualization, investigation, methodology, supervision, writing—original draft, writing—review and editing.

All authors gave final approval for publication and agreed to be held accountable for the work performed therein.

Conflict of interest declaration. We declare we have no competing interests.

Funding. No funding has been received for this article.

Acknowledgements. We thank Michael Dennis, Gary Marcus, Shane Legg, Yoshua Bengio, Jürgen Schmidhuber, Robert Kirk, Laura Ruis, Akbir Khan and Jakob Foerster for feedback on earlier drafts of this work.

References

1. O'Brien WJ, Browman HI, Evans BI. 1990 Search strategies of foraging animals. *Am. Sci.* **78**, 152–160.
2. Gordon DM. 1999 *Ants at work: how an insect society is organized*. New York, NY: Simon and Schuster.
3. Manser MB, Bell MB. 2004 Spatial representation of shelter locations in meerkats, *Suricata suricatta*. *Anim. Behav.* **68**, 151–157. (doi:10.1016/j.anbehav.2003.10.017)
4. Piaget J. 1952 *The origins of intelligence in children*. New York, NY: W. W. Norton & Co.
5. Piaget J. 1954 *The construction of reality in the child*. New York, NY: Basic Books.
6. Spodek B, Saracho ON. 2014 *Handbook of research on the education of young children*. New York, NY: Routledge.
7. Johnson JE, Sevimli-Celik S, Al-Mansour MA, Tunçdemir TBA, Dong PI. 2019 Play in early childhood education. In *Handbook of research on the education of young children*, pp. 165–175. New York, NY; Oxford, UK: Routledge.
8. Yawkey TD, Pellegrini AD. 2018 *Child's play: developmental and applied*, vol. 20. Oxford, UK: Routledge.
9. Andersen MM, Kiverstein J, Miller M, Roepstorff A. 2022 Play in predictive minds: a cognitive theory of play. *Psychol. Rev.* **130**, 462–479. (doi:10.1037/rev0000369)
10. Schmidhuber J. 2007 Gödel machines: fully self-referential optimal universal self-improvers. In *Artificial general intelligence*, pp. 199–226. Berlin, Germany: Springer.
11. Hutter M. 2007 Universal algorithmic intelligence: a mathematical top → down approach. In *Artificial general intelligence*, pp. 227–290. Berlin, Germany: Springer.
12. Brown T *et al.* 2020 Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **33**, 1877–1901.
13. Schrittwieser J *et al.* 2020 Mastering atari, go, chess and shogi by planning with a learned model. *Nature* **588**, 604–609. (doi:10.1038/s41586-020-03051-4)
14. Stooke A *et al.* 2021 Open-ended learning leads to generally capable agents. Preprint. (https://arxiv.org/abs/2107.12808)
15. Vygotsky LS, Cole M. 1978 *Mind in society: development of higher psychological processes*. Cambridge, UK: Harvard University Press.

16. Wood D, Bruner JS, Ross G. 1976 The role of tutoring in problem solving. *J. Child Psychol. Psychiatry* **17**, 89–100. (doi:10.1111/jcpp.1976.17.issue-2)

17. Hoffmann J *et al.* 2022 Training compute-optimal large language models. Preprint. (https://arxiv.org/abs/2203.15556)

18. Chowdhery A *et al.* 2022 Palm: scaling language modeling with pathways. Preprint. (https://arxiv.org/abs/2204.02311)

19. Ramesh A, Dhariwal P, Nichol A, Chu C, Chen M. 2022 Hierarchical text-conditional image generation with clip latents. Preprint. (https://arxiv.org/abs/2204.06125)

20. Alayrac JB *et al.* 2022 Flamingo: a visual language model for few-shot learning. Preprint. (https://arxiv.org/abs/2204.14198)

21. Ho J, Salimans T, Gritsenko A, Chan W, Norouzi M, Fleet DJ. 2022 Video diffusion models. Preprint. (https://arxiv.org/abs/2204.03458)

22. Oord A, Li Y, Vinyals O. 2018 Representation learning with contrastive predictive coding. Preprint. (https://arxiv.org/abs/1807.03748)

23. Sutton RS, Barto AG. 2018 *Reinforcement learning: an introduction*. New York, NY: MIT Press.

24. Bellemare MG, Naddaf Y, Veness J, Bowling M. 2013 The arcade learning environment: an evaluation platform for general agents. *J. Artif. Intell. Res.* **47**, 253–279. (doi:10.1613/jair.3912)

25. Küttler H, Nardelli N, Miller A, Raileanu R, Selvatici M, Grefenstette E, Rocktäschel T. 2020 The nethack learning environment. *Adv. Neural Inf. Process. Syst.* **33**, 7671–7684.

26. Hadsell R, Rao D, Rusu AA, Pascanu R. 2020 Embracing change: continual learning in deep neural networks. *Trends Cogn. Sci*. **24**, 1028–1040. (doi:10.1016/j.tics.2020.09.004)

27. Zhu X, Vondrick C, Fowlkes CC, Ramanan D. 2016 Do we need more training data? *Int. J. Comput. Vision* **119**, 76–92. (doi:10.1007/s11263-015-0812-2)

28. Paul M, Ganguli S, Dziugaite GK. 2021 Deep learning on a data diet: finding important examples early in training. *Adv. Neural Inf. Process. Syst.* **34**, 20 596–20 5607.

29. Sorscher B, Geirhos R, Shekhar S, Ganguli S, Morcos AS. 2022 Beyond neural scaling laws: beating power law scaling via data pruning. *Adv. Neural Inf. Process. Syst.* **36**.

30. Moore G. 1965 Moore's law. *Electron. Mag.* **38**, 114. (doi:10.2307/3756714)

31. Shalf J. 2020 The future of computing beyond Moore's law. *Philos. Trans. R. Soc. A* **378**, 20190061. (doi:10.1098/rsta.2019.0061)

32. Schmidhuber J. 2010 Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE Trans. Auton. Ment. Dev.* **2**, 230–247. (doi:10.1109/TAMD.2010.2056368)

33. Schmidhuber J. 2013 Powerplay: training an increasingly general problem solver by continually searching for the simplest still unsolvable problem. *Front. Psychol.* **4**, 313. (doi:10.3389/fpsyg.2013.00313)

34. Clune J. 2019 AI-GAs: AI-generating algorithms, an alternate paradigm for producing general artificial intelligence. Preprint. (https://arxiv.org/abs/1905.10985)

35. Lehman J, Stanley KO. 2011 Abandoning objectives: evolution through the search for novelty alone. *Evol. Comput.* **19**, 189–223. (doi:10.1162/EVCO_a_00025)

36. Stanley KO, Lehman J. 2015 *Why greatness cannot be planned: the myth of the objective*. Berlin, Germany: Springer.

37. Stanley KO, Lehman J, Soros L. 2017 Open-endedness: the last grand challenge you've never heard of. While open-endedness could be a force for discovering intelligence, it could also be a component of AI itself.

38. Karras T, Laine S, Aila T. 2019 A style-based generator architecture for generative adversarial networks. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, *Long Beach, CA, 16–20 June 2019*, pp. 4401–4410. New York, NY: IEEE.

39. van den Oord A, Dieleman S, Zen H, Simonyan K, Vinyals O, Graves A, Kalchbrenner N, Senior A, Kavukcuoglu K. 2016 WaveNet: a generative model for raw audio. In *The 9th ISCA Speech Synthesis Workshop, Sunnyvale, CA, USA, 13–15 September 2016*, pp. 125. Baixas, Franc: ISCA.

40. Shen J *et al.* 2018 Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, *Calgary, Canada, 15–20 April*, pp. 4779–4783. New York, NY: IEEE.

41. Silver D *et al.* 2016 Mastering the game of Go with deep neural networks and tree search. *Nature* **529**, 484–489. (doi:10.1038/nature16961)

42. Silver D *et al.* 2017 Mastering chess and shogi by self-play with a general reinforcement learning algorithm. Preprint. (https://arxiv.org/abs/1712.01815)

43. Perolat J *et al.* 2022 Mastering the game of stratego with model-free multiagent reinforcement learning. E-prints (http://arxiv.org/abs/2206)

44. Mirhoseini A *et al.* 2021 A graph placement methodology for fast chip design. *Nature* **594**, 207–212. (doi:10.1038/s41586-021-03544-w)

45. Roy R, Raiman J, Kant N, Elkin I, Kirby R, Siu M, Oberman S, Godil S, Catanzaro B. 2021 Prefixrl: optimization of parallel prefix circuits using deep reinforcement learning. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, *Phoenix, AZ, 5–9 December 2021, USA*, pp. 853–858. New York NY: IEEE.

46. Degrave J *et al.* 2022 Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature* **602**, 414–419. (doi:10.1038/s41586-021-04301-9)

47. Marcus G. 2020 The next decade in AI: four steps towards robust artificial intelligence. Preprint. (https://arxiv.org/abs/2002.06177)

48. Pearl J. 2009 *Causality*. Cambridge, UK: Cambridge University Press.

49. Ortega PA *et al.* 2021 Shaking the foundations: delusions in sequence models for interaction and control. Preprint. (https://arxiv.org/abs/2110.10819)

50. LeCun Y. 2022 A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. (https://openreview.net/pdf?id=BZ5a1r-kVsf)

51. Boyd R, Richerson PJ, Henrich J. 2011 The cultural niche: why social learning is essential for human adaptation. *Proc. Natl Acad. Sci. USA* **108**, 10 918–10 925. (doi:10.1073/pnas.1100290108)

52. Henrich J. 2015 The secret of our success. In *The secret of our success*. Princeton, NJ: Princeton University Press.

53. Ha D, Tang Y. 2022 Collective intelligence for deep learning: a survey of recent developments. *Collect. Intell.* **1**, 263391372211148. (doi:10.1177/26339137221114874)

54. Cubuk ED, Zoph B, Mane D, Vasudevan V, Le QV. 2019 Autoaugment: learning augmentation strategies from data. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, *Long Beach, CA, 16–20 June 2019*, pp. 113–123. New York, NY: IEEE.

55. Cobbe K, Klimov O, Hesse C, Kim T, Schulman J. 2019 Quantifying generalization in reinforcement learning. In *Int. Conf. on Machine Learning*, *Long Beach, CA, 10–15 June 2019*, pp. 1282–1289. PMLR.

56. Kaplan J *et al.* 2020 Scaling laws for neural language models. Preprint. (https://arxiv.org/abs/2001.08361)

57. Yarats D, Brandfonbrener D, Liu H, Laskin M, Abbeel P, Lazaric A, Pinto L. 2022 Don't change the algorithm, change the data: exploratory data for offline reinforcement learning. Preprint. (https://arxiv.org/abs/2201.13425)

58. Klein PD. 1999 Human knowledge and the infinite regress of reasons. *Philos. Perspect.* **13**, 297–325. (doi:10.1111/0029-4624.33.s13.14)

59. Bellman R. 1966 Dynamic programming. *Science* **153**, 34–37. (doi:10.1126/science.153.3731.34)

60. Petrenko A, Huang Z, Kumar T, Sukhatme G, Koltun V. 2020 Sample factory: egocentric 3D control from pixels at 100 000 fps with asynchronous reinforcement learning. In *Int. Conf. on Machine Learning*, *Vienna, Austria, 13–18 July 2020*, pp. 7652–7662. PMLR.

61. Freeman CD, Frey E, Raichuk A, Girgin S, Mordatch I, Bachem O. 2021 Brax—a differentiable physics engine for large scale rigid body simulation. Preprint. (https://arxiv.org/abs/2106.13281)

62. Chan SC, Santoro A, Lampinen AK, Wang JX, Singh A, Richemond PH, McClelland J, Hill F. 2022 Data distributional properties drive emergent in-context learning in transformers. Preprint. (https://arxiv.org/abs/2205.05055)

63. Williams CK, Rasmussen CE. 2006 *Gaussian processes for machine learning*, vol. 2. Cambridge, MA: MIT Press.

64. Kendall A, Gal Y. 2017 What uncertainties do we need in bayesian deep learning for computer vision? *Adv. Neural. Inf. Process. Syst* **30**, 5574–5584.

65. Clements WR, Van Delft B, Robaglia BM, Slaoui RB, Toth S. 2019 Estimating risk and uncertainty in deep reinforcement learning. Preprint. (https://arxiv.org/abs/1905.09638)

66. Hüllermeier E, Waegeman W. 2021 Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. *Mach. Learn.* **110**, 457–506. (doi:10.1007/s10994-021-05946-3)

67. Dennis M, Jaques N, Vinitsky E, Bayen A, Russell S, Critch A, Levine S. 2020 Emergent complexity and zero-shot transfer via unsupervised

environment design. *Adv. Neural Inf. Process. Syst.* **33**, 13 049–13 061.

68. Graves A, Bellemare MG, Menick J, Munos R, Kavukcuoglu K. 2017 Automated curriculum learning for neural networks. In *Int. Conf. on Machine Learning*, *Sydney, Australia, 6–11 August 2017*, pp. 1311–1320. PMLR.

69. Settles B. 2009 Active learning literature survey. Madison, WI: University of Wisconsin-Madison Department of Computer Sciences.

70. Cohn D, Atlas L, Ladner R. 1994 Improving generalization with active learning. *Mach. Learn.* **15**, 201–221. (doi:10.1007/BF00993277)

71. Schaul T, Quan J, Antonoglou I, Silver D. 2016 Prioritized experience replay. *Int. Conf. on Learning Representations, San Juan, Puerto Rico, 2–4 May 2016*. Conference Track Proceedings. (http://arxiv.org/abs/1511.05952)

72. Fails JA, Olsen Jr DR. 2003 Interactive machine learning. In *Proc. of the 8th Int. Conf. on Intelligent User Interfaces*, *Miami, FL, 12–15 January 2003*, pp. 39–45. New York, NY: ACM.

73. Richalet J, Rault A, Testud J, Papon J. 1978 Model predictive heuristic control: applications to industrial processes. *Automatica* **14**, 413–428. (doi:10.1016/0005-1098(78)90001-8)

74. Hafner D, Lillicrap TP, Ba J, Norouzi M. 2020 Dream to control: learning behaviors by latent imagination. *Int. Conf. on Learning Representations, Addis Ababa, 26–30 April*. Conference Track Proceedings. See https://openreview.net/forum?id=S1lOTC4tDS.

75. Hafner D, Lillicrap TP, Norouzi M, Ba J. 2021 Mastering atari with discrete world models. *Int. Conf. on Learning Representations, Virtual, May 3–7 2021*. OpenReview.net. See https://openreview.net/forum?id=0oabwyZbOu.

76. Helmbold DP, Littlestone N, Long PM. 1992 Apple tasting and nearly one-sided learning. In *Proc. 33rd Annual Symp. on Foundations of Computer Science*, *Pittsburgh, PA, 24–27 October 1992*, pp. 493–502. New York, NY: IEEE Computer Society.

77. Jiang H, Jiang Q, Pacchiano A. 2021 Learning the truth from only one side of the story. In *Int. Conf. on Artificial Intelligence and Statistics*, *Virtual, 13–15 April 2021*, pp. 2413–2421. PMLR.

78. Hardt M, Megiddo N, Papadimitriou C, Wootters M. 2016 Strategic classification. In *Proc. of the 2016 ACM Conf. on Innovations in Theoretical Computer Science*, *Cambridge, MA, 14–17 January 2016*, pp. 111–122. New York, NY: ACM.

79. Perdomo J, Zrnic T, Mendler-Dünner C, Hardt M. 2020 Performative prediction. In *Int. Conf. on Machine Learning*, *Vienna, Austria, 13–18 July 2020*, pp. 7599–7609. PMLR.

80. Bellogín A, Castells P, Cantador I. 2017 Statistical biases in information retrieval metrics for recommender systems. *Inf. Retr. J.* **20**, 606–634. (doi:10.1007/s10791-017-9312-z)

81. Ge Y, Zhao S, Zhou H, Pei C, Sun F, Ou W, Zhang Y, 2020 Understanding echo chambers in e-commerce recommender systems. In *Proc. of the 43rd Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, *Virtual, 25–30 July*, pp. 2261–2270. New York, NY: ACM.

82. Neophytou N, Mitra B, Stinson C. 2022 Revisiting popularity and demographic biases in recommender evaluation and effectiveness. In *European Conf. on Information Retrieval*, *Stavanger, Norway, 10–14 April 2022*, pp. 641–654. Berlin, Germany: Springer.

83. Karpathy A. 2021 AI for full self-driving. In *Conf. on Computer Vision and Pattern Recognition*. See https://www.youtube.com/watch?v=g6bOwQdCJrc&ab_channel=WADatCVPR.

84. Karpathy A. 2019 Tesla autonomy day. See https://www.youtube.com/watch?v=Ucp0TTmvqOE&ab_channel=Tesla.

85. Bartlett R, Morse A, Stanton R, Wallace N. 2022 Consumer-lending discrimination in the FinTech era. *J. Financ. Econ.* **143**, 30–56. (doi:10.1016/j.jfineco.2021.05.047)

86. Hong S, Kim SH. 2016 Political polarization on twitter: implications for the use of social media in digital governments. *Gov. Inf. Q.* **33**, 777–782. (doi:10.1016/j.giq.2016.04.007)

87. Tucker JA, Guess A, Barberá P, Vaccari C, Siegel A, Sanovich S, Stukal D, Sanovich S, Nyhan B. 2018 Social media, political polarization, and political disinformation: a review of the scientific literature. Political polarization, and political disinformation: a review of the scientific literature (19 March 2018)

88. Bak-Coleman JB *et al.* 2021 Stewardship of global collective behavior. *Proc. Natl Acad. Sci. USA* **118**, e2025764118. (doi:10.1073/pnas.2025764118)

89. Amin S, Gomrokchi M, Satija H, van Hoof H, Precup D. 2021 A survey of exploration methods in reinforcement learning. Preprint. (https://arxiv.org/abs/2109.00157)

90. Ostrovski G, Bellemare MG, Oord A, Munos R. 2017 Count-based exploration with neural density models. In *Int. Conf. on Machine Learning*, *Sydney, Australia, 6–11 August 2017*, pp. 2721–2730. PMLR.

91. Pathak D, Gandhi D, Guptaandhi A, 2019 Self-supervised exploration via disagreement. In *Int. Conf. on Machine Learning*, *Long Beach, CA, 10–15 June 2019*, pp. 5062–5071. PMLR.

92. Sekar R, Rybkin O, Daniilidis K, Abbeel P, Hafner D, Pathak D. 2020 Planning to explore via self-supervised world models. In *Int. Conf. on Machine Learning*, *Vienna, Austria, 13–18 July 2020*, pp. 8583–8592. PMLR.

93. Oudeyer PY, Kaplan F, Hafner VV. 2007 Intrinsic motivation systems for autonomous mental development. *IEEE Trans. Evol. Comput.* **11**, 265–286. (doi:10.1109/TEVC.2006.890271)

94. Kompella VR, Stollenga M, Luciw M, Schmidhuber J. 2017 Continual curiosity-driven skill acquisition from high-dimensional video inputs for humanoid robots. *Artif. Intell.* **247**, 313–335. (doi:10.1016/j.artint.2015.02.001)

95. Der Kiureghian A, Ditlevsen O. 2009 Aleatory or epistemic? Does it matter? *Struct. Saf.* **31**, 105–112.

96. Mavor-Parker A, Young K, Barry C, Griffin L. 2022 How to stay curious while avoiding noisy TVs using aleatoric uncertainty estimation. In *Int. Conf. on Machine Learning*, *Baltimore, MD, 17–23 July 2022*, pp. 15 220–15 240. PMLR.

97. Singh S, Lewis RL, Barto AG, Sorg J. 2010 Intrinsically motivated reinforcement learning: an evolutionary perspective. *IEEE Trans. Auton. Ment. Dev.* **2**, 70–82. (doi:10.1109/TAMD.2010.2051031)

98. Munos R, Stepleton T, Harutyunyan A, Bellemare M. 2016 Safe and efficient off-policy reinforcement learning. *Adv. Neural Inf. Process. Syst.* **29**, 1046–1054.

99. Pathak D, Agrawal P, Efros AA, Darrell T. 2017 Curiosity-driven exploration by self-supervised prediction. In *Int. Conf. on Machine Learning*, *Sydney, Australia, 6–11 August 2017*, pp. 2778–2787. PMLR.

100. Burda Y, Edwards H, Storkey AJ, Klimov O. 2019 Exploration by random network distillation. *Int. Conf. on Learning Representations, New Orleans, LA, 6–9 May 2019*. OpenReview.net. See https://openreview.net/forum?id=H1lJJnR5Ym.

101. Raileanu R, Rocktäschel T. 2020 RIDE: rewarding impact-driven exploration for procedurally-generated environments. *Int. Conf. on Learning Representations, Addis Ababa, Ethiopia, 26–30 April 2020*. Conference Track Proceedings. (https://openreview.net/forum?id=rkg-TJBFPB)

102. Zhang T, Xu H, Wang X, Wu Y, Keutzer K, Gonzalez JE, Tian Y. 2021 NovelD: a simple yet effective exploration criterion. *Adv. Neural Inf. Process. Syst.* **34**, 25 217–25 230. (doi:10.1016/j.strusafe.2008.06.020)

103. Henaff M, Raileanu R, Jiang M, Rocktäschel T. 2022 Exploration via elliptical episodic bonuses. *Adv. Neural Inf. Process. Syst.* **36**.

104. Singh S, Jaakkola T, Littman ML, Szepesvári C. 2000 Convergence results for single-step on-policy reinforcement-learning algorithms. *Mach. Learn.* **38**, 287–308. (doi:10.1023/A:1007678930559)

105. Thompson WR. 1933 On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* **25**, 285–294. (doi:10.1093/biomet/25.3-4.285)

106. Osband I, Blundell C, Pritzel A, Van Roy B. 2016 Deep exploration via bootstrapped DQN. *Adv. Neural Inf. Process. Syst.* **29**, 4026–4034.

107. Risi S, Togelius J. 2020 Increasing generality in machine learning through procedural content generation. *Nat. Mach. Intell.* **2**, 428–436. (doi:10.1038/s42256-020-0208-z)

108. Portelas R, Colas C, Weng L, Hofmann K, Oudeyer PY. 2020 Automatic curriculum learning for deep rl: a short survey. Preprint. (https://arxiv.org/abs/2003.046640)

109. Peterson M. 2017 *An introduction to decision theory*. Cambridge, UK: Cambridge University Press.

110. Nash JF. 1950 Equilibrium points in *n*-person games. *Proc. Natl Acad. Sci. USA* **36**, 48–49. (doi:10.1073/pnas.36.1.48)

111. Savage LJ. 1951 The theory of statistical decision. *J. Am. Stat. Assoc.* **46**, 55–67. (doi:10.1080/01621459.1951.10500768)

112. Pugh JK, Soros LB, Stanley KO. 2016 Quality diversity: a new frontier for evolutionary computation. *Front. Rob. AI* **3**, 40.

113. Houlsby N, Huszár F, Ghahramani Z, Lengyel M. 2011 Bayesian active learning for classification and preference learning. Preprint. (https://arxiv.org/abs/1112.5745)

114. Mindermann S *et al.* 2022 Prioritized training on points that are learnable, worth learning,

and not yet learnt. In *Int. Conf. on Machine Learning*, *Baltimore, MD, 17–23 July 2022*, pp. 15 630–15 649. PMLR.

115. Zhang H, Cisse M, Dauphin YN, Lopez-Paz D. 2017 Mixup: beyond empirical risk minimization. Preprint. (https://arxiv.org/abs/1710.09412)
116. Jahanian A, Puig X, Tian Y, Isola P. 2021 Generative models as a data source for multiview representation learning. Preprint. (https://arxiv.org/abs/2106.05258)
117. Von Kügelgen J, Sharma Y, Gresele L, Brendel W, Schölkopf B, Besserve M, Locatello F. 2021 Self-supervised learning with data augmentations provably isolates content from style. *Adv. Neural Inf. Process. Syst.* **34**, 16 451–16 467.
118. Ilse M, Tomczak JM, Forré P. 2021 Selecting data augmentation for simulating interventions. In *Int. Conf. on Machine Learning*, *Virtual, 18–24 July 2021*, pp. 4555–4562. PMLR.
119. Mao C, Cha A, Gupta A, Wang H, Yang J, Vondrick C. 2021 Generative interventions for causal learning. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, *Montreal, Canada, 20–25 June 2021*, pp. 3947–3956. New York, NY: IEEE.
120. Sohl-Dickstein J, Weiss E, Maheswaranathan N, Ganguli S. 2015 Deep unsupervised learning using nonequilibrium thermodynamics. In *Int. Conf. on Machine Learning*, *Lille, France, 6–11 July 2015*, pp. 2256–2265. PMLR.
121. Song Y, Durkan C, Murray I, Ermon S. 2021 Maximum likelihood training of score-based diffusion models. *Adv. Neural Inf. Process. Syst.* **34**, 1415–1428.
122. Kingma D, Salimans T, Poole B, Ho J. 2021 Variational diffusion models. *Adv. Neural Inf. Process. Syst.* **34**, 21 696–21 707.
123. Nichol AQ, Dhariwal P, Ramesh A, Shyam P, Mishkin P, McGrew B, Sutskever I, Chen M. 2022 GLIDE: towards photorealistic image generation and editing with text-guided diffusion models. In *Int. Conf. on Machine Learning*, *Baltimore, MD, 17–23 July 2022*, pp. 16784–16804. PMLR.
124. Christiano PF, Leike J, Brown T, Martic M, Legg S, Amodei D. 2017 Deep reinforcement learning from human preferences. *Adv. Neural Inf. Process. Syst.* **30**, 4299–4307.
125. Ouyang L *et al.* 2022 Training language models to follow instructions with human feedback. Preprint. (https://arxiv.org/abs/2203.02155)
126. Thoppilan R *et al.* 2022 LaMDA: language models for dialog applications. Preprint. (https://arxiv.org/abs/2201.08239)
127. Shi T, Karpathy A, Fan L, Hernandez J, Liang P. 2017 World of bits: an open-domain platform for web-based agents. In *Int. Conf. on Machine Learning*, *Sydney, Australia, 6–11 August 2017*, pp. 3135–3144. PMLR.
128. Gur I, Jaques N, Miao Y, Choi J, Tiwari M, Lee H, Faust A. 2021 Environment generation for zero-shot compositional reinforcement learning. *Adv. Neural Inf. Process. Syst.* **34**, 4157–4169.
129. Nakano R *et al.* 2021 WebGPT: browser-assisted question-answering with human feedback. CoRR (https://arxiv.org/abs/2112.09332)
130. Baker B, Akkaya I, Zhokhov P, Huizinga J, Tang J, Ecoffet A, Houghton B, Sampedro R, Clune J. 2022 Video PreTraining (VPT): learning to act by watching unlabeled online videos. Preprint. (https://arxiv.org/abs/2206.11795)
131. Kirsch A, Van Amersfoort J, Gal Y. 2019 BatchBALD: efficient and diverse batch acquisition for deep bayesian active learning. *Adv. Neural Inf. Process. Syst.* **32**, 7024–7035.
132. Jiang M, Dennis M, Parker-Holder J, Foerster J, Grefenstette E, Rocktäschel T. 2021 Replay-guided adversarial environment design. *Adv. Neural Inf. Process. Syst.* **35**, 1884–1897.
133. Parker-Holder J, Jiang M, Dennis M, Samvelyan M, Foerster J, Grefenstette E, Rocktäschel T. 2022 Evolving curricula with regret-based environment design. In *Int. Conf. on Machine Learning*, *Baltimore, MD, 17–23 July 2022*, pp. 17473–17498. PMLR.
134. Baluja S, Seth R, Sivakumar D, Jing Y, Yagnik J, Kumar S, Ravichandran D, Aly M. 2008 Video suggestion and discovery for youtube: taking random walks through the view graph. In *Proc. of the 17th Int. Conf. on World Wide Web*, *Beijing, China, 21–25 April*, pp. 895–904. New York, NY: ACM.
135. Hu X, Binaykiya T, Frank E, Cirit O. 2022 DeeprETA: an ETA post-processing system at scale. Preprint. (https://arxiv.org/abs/2206.02127)
136. Nakkiran P, Kaplun G, Bansal Y, Yang T, Barak B, Sutskever I. 2021 Deep double descent: where bigger models and more data hurt. *J. Stat. Mech: Theory Exp.* **2021**, 124003. (doi:10.1088/1742-5468/ac3a74)
137. Ratner AJ, De Sa CM, Wu S, Selsam D, Ré C. 2016 Data programming: creating large training sets, quickly. *Adv. Neural Inf. Process. Syst.* **29**, 3567–3575.
138. Ratner AJ, Bach SH, Ehrenberg HR, Ré C. 2017 Snorkel: fast training set generation for information extraction. In *Proc. of the 2017 ACM Int. Conf. on Management of Data*, *Chicago, IL, 14–19 May*, pp. 1683–1686. New York, NY: ACM.
139. Karpathy A. Medium, editor. Software 2.0. See https://karpathy.medium.com/software-2-0-a64152b37c35.
140. Carbin M. 2019 Overparameterization: a connection between software 1.0 and software 2.0. In *3rd Summit on Advances in Programming Languages (SNAPL 2019)* (eds BS Lerner, R Bodík, S Krishnamurthi), Leibniz International Proceedings in Informatics (LIPIcs), vol. 136, pp. 1:1–1:13. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. See http://drops.dagstuhl.de/opus/volltexte/2019/10544.
141. Czarnecki WM, Gidel G, Tracey B, Tuyls K, Omidshafiei S, Balduzzi D, Jaderberg M. 2020 Real world games look like spinning tops. *Adv. Neural Inf. Process. Syst.* **33**, 17 443–17 454.
142. Legg S, Hutter M. 2007 Universal intelligence: a definition of machine intelligence. *Minds Mach.* **17**, 391–444. (doi:10.1007/s11023-007-9079-x)
143. Veness J, Ng KS, Hutter M, Uther W, Silver D. 2011 A monte-carlo aixi approximation. *J. Artif. Intell. Res.* **40**, 95–142. (doi:10.1613/jair.3125)
144. Mitchell T *et al.* 2018 Never-ending learning. *Commun. ACM* **61**, 103–115. (doi:10.1145/3191513)
145. Boden MA. 1996 *The philosophy of artificial life*. Oxford, UK: Oxford University Press.
146. Langton CG. 1997 *Artificial life: an overview*. Cambridge, MA: MIT Press.
147. Jiang M, Grefenstette E, Rocktäschel T. 2021 Prioritized level replay. In *Proc. of the 38th Int. Conf. on Machine Learning, Int. Conf. on Machine Learning, Virtual, 18–24 July 2021*, pp. 4940–4950. PMLR. See http://proceedings.mlr.press/v139/jiang21b.html.
148. Wang A, Singh A, Michael J, Hill F, Levy O, Bowman SR. 2019 GLUE: a multi-task benchmark and analysis platform for natural language understanding. *Int. Conf. on Learning Representations*, *New Orleans, LA, 6–9 May 2019*. Conference Track Proceedings. See https://openreview.net/forum?id=rJ4km2R5t7.
149. Wang A, Pruksachatkun Y, Nangia N, Singh A, Michael J, Hill F, Levy O, Bowman S. 2019 SuperGLUE: a stickier benchmark for general-purpose language understanding systems. *Adv. Neural Inf. Process. Syst.* **32**, 3261–3275.
150. Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L. 2009 Imagenet: a large-scale hierarchical image database. In *IEEE Conf. on Computer Vision and Pattern Recognition*, *Miami, FL, 20–25 June 2009*, pp. 248–255. New York, NY: IEEE.
151. Liška A *et al.* 2022 StreamingQA: a benchmark for adaptation to new knowledge over time in question answering models. Preprint. (https://arxiv.org/abs/2205.11388)
152. Srivastava A *et al.* 2022 Beyond the imitation game: quantifying and extrapolating the capabilities of language models. Preprint. (https://arxiv.org/abs/2206.04615)
153. Kiela D *et al.* 2021 Dynabench: rethinking benchmarking in NLP. In *Proc. of the 2021 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Virtual, 6–11 June 2021*, pp. 4110–4124. New York, NY: ACM.
154. Tessler C, Givony S, Zahavy T, Mankowitz D, Mannor S. 2017 A deep hierarchical approach to lifelong learning in minecraft. In *Proc. of the AAAI Conf. on Artificial Intelligence*, *San Francisco, CA, 4–9 February 2017*. New York, NY: ACM.
155. Guss WH, Houghton B, Topin N, Wang P, Codel C, Veloso M, Salakhutdinov R. 2019 MineRL: a large-scale dataset of minecraft demonstrations. In *Proc. of the Twenty-Eighth International Joint Conf. on Artificial Intelligence, Macao, China, 10–16 August 2019*, pp. 2442–2448. New York, NY: ACM.
156. Grbic D, Palm RB, Najarro E, Glanois C, Risi S. 2021 EvoCraft: a new challenge for open-endedness. In *Int. Conf. on the Applications of Evolutionary Computation (Part of EvoStar)*, *Seville, Spain 7–9 April*, pp. 325–340. Berlin, Germany: Springer.
157. Fan L *et al.* 2022 MineDojo: building open-ended embodied agents with internet-scale knowledge. Preprint. (https://arxiv.org/abs/2206.08853)
158. Samvelyan M *et al.* 2021 MiniHack the planet: a sandbox for open-ended reinforcement learning research. In *Proc. of the Neural Information Processing Systems Track on Datasets and Benchmarks, Virtual,*

6–14 December 2021. Conference Track Proceedings. (arXiv:2109.13202v2)

159. Lehman J, Gordon J, Jain S, Ndousse K, Yeh C, Stanley KO. 2022 Evolution through large models. Preprint. (https://arxiv.org/abs/2206.08896)
160. Bedau MA, Snyder E, Packard NH. 1998 A classification of long-term evolutionary dynamics. In *ALIFE: Proceedings of the sixth international conference on artificial life*, *Los Angeles, CA, 26–29 June 1998*, pp. 228–237. New York, NY: ACM.
161. Standish RK. 2003 Open-ended artificial evolution. *Int. J. Comput. Intell. Appl.* **3**, 167–175. (doi:10.1142/S1469026803000914)
162. Soros L, Stanley K. 2014 Identifying necessary conditions for open-ended evolution through the artificial life world of chromaria. In *ALIFE 14: The Fourteenth Int. Conf. on the Synthesis and Simulation of Living Systems*, *New York, NY, 30 July–2 August 2014*, pp. 793–800. Cambridge, MA: MIT Press.
163. Wang R, Lehman J, Rawal A, Zhi J, Li Y, Clune J, Stanley K. 2020 Enhanced POET: open-ended reinforcement learning through unbounded invention of learning challenges and their solutions. In *Int. Conf. on Machine Learning*, *Vienna, Austria, 12–18 July 2020*, pp. 9940–9951. PMLR.
164. Tran T, Do TT, Reid I, Carneiro G. 2019 Bayesian generative active deep learning. In *Int. Conf. on Machine Learning*, *Long Beach, CA, 10–15 June 2019*, pp. 6295–6304. PMLR.
165. Kusner MJ, Paige B, Hernández-Lobato JM. 2017 Grammar variational autoencoder. In *International Conf. on Machine Learning*, *Sydney, Australia, 6–11 August 2017*, pp. 1945–1954. PMLR.
166. Lu X, Gonzalez J, Dai Z, Lawrence ND. 2018 Structured variationally auto-encoded optimization. In *Int. Conf. on Machine Learning*, *Stockhold, Sweden, 10–15 June 2018*, pp. 3267–3275. PMLR.
167. Winter R, Montanari F, Steffen A, Briem H, Noé F, Clevert DA. 2019 Efficient multi-objective molecular optimization in a continuous latent space. *Chem. Sci.* **10**, 8016–8024. (doi:10.1039/C9SC01928F)
168. Tripp A, Daxberger E, Hernández-Lobato JM. 2020 Sample-efficient optimization in the latent space of deep generative models via weighted retraining. *Adv. Neural Inf. Process. Syst.* **33**, 11 259–11 272.
169. Gaier A, Asteroth A, Mouret JB. 2018 Data-efficient design exploration through surrogate-assisted illumination. *Evol. Comput.* **26**, 381–410. (doi:10.1162/evco_a_00231)
170. Zhang Y, Fontaine MC, Hoover AK, Nikolaidis S. 2022 Deep surrogate assisted MAP-elites for automated hearthstone deckbuilding. In *Proc. of the Genetic and Evolutionary Computation Conference*, *Lille, France, 10–14 July 2022*, pp. 158–167. New York, NY: ACM.
171. Bhatt V, Tjanaka B, Fontaine MC, Nikolaidis S. 2022 Deep surrogate assisted generation of environments. *Adv. Neural Inf. Process. Syst*. **35**.
172. Petrenko A, Wijmans E, Shacklett B, Koltun V. 2021 Megaverse: simulating embodied agents at one million experiences per second. In *Int. Conf. on Machine Learning*, *Virtual, 18–24 July 2021*, pp. 8556–8566. PMLR.
173. Jaderberg M, Simonyan K, Vedaldi A, Zisserman A. 2014 Synthetic data and artificial neural networks for natural scene text recognition. Preprint. (https://arxiv.org/abs/1406.2227)
174. Bansal M, Krizhevsky A, Ogale AS. 2019 ChauffeurNet: learning to drive by imitating the best and synthesizing the worst. In *Robotics: Science and Systems*, *Freiburg im Breisgau, Germany, 22–26 June*. Conference Track Proceedings.
175. Tripathi S, Chandra S, Agrawal A, Tyagi A, Rehg JM, Chari V. 2019 Learning to generate synthetic data via compositing. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, *Long Beach, CA, 16–20 June 2019*, pp. 461–470. New York, NY: IEEE.
176. Such FP, Rawal A, Lehman J, Stanley K, Clune J. 2020 Generative teaching networks: accelerating neural architecture search by learning to generate synthetic training data. In *Int. Conf. on Machine Learning*, *Virtual, 12–18 June 2020*, pp. 9206–9216. PMLR.
177. Bouchard G, Stenetorp P, Riedel S. 2016 Learning to generate textual data. In *Proc. of the 2016 Conf. on Empirical Methods in Natural Language Processing*, *Austin, TX, 1–5 November 2016*, pp. 1608–1616. Cambridge, MA: MIT Press.
178. Nye M, Solar-Lezama A, Tenenbaum J, Lake BM. 2020 Learning compositional rules via neural program synthesis. *Adv. Neural Inf. Process. Syst.* **33**, 10 832–10 842.
179. Chen X, Liang C, Yu AW, Song D, Zhou D. 2020 Compositional generalization via neural-symbolic stack machines. *Adv. Neural Inf. Process. Syst.* **33**, 1690–1701.
180. Farquhar S, Gal Y, Rainforth T. 2021 On statistical bias in active learning: how and when to fix it. In *9th Int. Conf. on Learning Representations, Virtual, 3–7 May*. Conference Track Proceedings. See https://openreview.net/forum?id=JiYq3eqTKY.
181. Jiang M, Dennis MD, Parker-Holder J, Lupu A, Küttler H, Grefenstette E, Rocktäschel T, Foerster J. 2022 Grounding aleatoric uncertainty for unsupervised environment design. *Adv. Neural Inf. Process. Syst.* **36**.
182. Amodei D, Olah C, Steinhardt J, Christiano P, Schulman J, Mané D. 2016 Concrete problems in AI safety. Preprint. (https://arxiv.org/abs/1606.06565)
183. Ecoffet A, Clune J, Lehman J. 2020 Open questions in creating safe open-ended AI: tensions between control and creativity. Preprint. (https://arxiv.org/abs/2006.07495)
184. Robins A. 1995 Catastrophic forgetting, rehearsal and pseudorehearsal. *Connect. Sci.* **7**, 123–146. (doi:10.1080/09540099550039318)
185. French RM. 1999 Catastrophic forgetting in connectionist networks. *Trends Cogn. Sci.* **3**, 128–135. (doi:10.1016/S1364-6613(99)01294-2)
186. Kirkpatrick J *et al.* 2017 Overcoming catastrophic forgetting in neural networks. *Proc. Natl Acad. Sci. USA* **114**, 3521–3526. (doi:10.1073/pnas.1611835114)
187. Pascanu R, Mikolov T, Bengio Y. 2013 On the difficulty of training recurrent neural networks. In *Int. Conf. on Machine Learning*, *Atlanta, GA, 16–21 June 2013*, pp. 1310–1318. PMLR.
188. Schulman J, Heess N, Weber T, Abbeel P. 2015 Gradient estimation using stochastic computation graphs. *Adv. Neural Inf. Process. Syst.* **28**, 3528–3536.
189. Bottou L, Curtis FE, Nocedal J. 2018 Optimization methods for large-scale machine learning. *SIAM Rev.* **60**, 223–311. (doi:10.1137/16M1080173)
190. Balduzzi D, Racaniere S, Martens J, Foerster J, Tuyls K, Graepel T. 2018 The mechanics of *n*-player differentiable games. In *Int. Conf. on Machine Learning*, *Stockholm, Sweden, 10–15 July 2018*, pp. 354–363. PMLR.
191. Mazumdar EV, Jordan MI, Sastry SS. 2019 On finding local nash equilibria (and only local nash equilibria) in zero-sum games. Preprint. (https://arxiv.org/abs/1901.00838)
192. Mazumdar E, Ratliff LJ, Jordan MI, Sastry SS. 2019 Policy-gradient algorithms have no guarantees of convergence in linear quadratic games. Preprint. (https://arxiv.org/abs/1907.03712)