Deep Reinforcement Learning Methods For Scaling Active Inference*

Roland Dubb

University of Cape Town
Supervisors: Assoc. Prof. Jonathan Shock and Melusi Mavuso

March 2022

Abstract

Active inference provides a normative framework for agent-based decision-making in an uncertain environment. The active inference agent performs variational Bayesian inference which unites decision-making, learning and perceptual inference. These are united via the minimisation of the agent's free energy objective function. This objective function embeds the desires of the agent into a prior preference distribution. The objective function is also furnished with a directed intrinsic exploratory motive which provides a compelling means for the agent to tackle the exploration-exploitation dilemma. However, many early attempts at constructing active inference agents have been limited to tabular approaches. As both reinforcement learning and active inference agents address the problem of agent-based decision-making in uncertain environments, to achieve some goal, it is appropriate to turn to the success of deep reinforcement learning as a means of addressing the problem of scaling the active inference agents. Particularly, the success (of deep reinforcement learning agents) in scaling tabular methods to tackle larger state-spaces and longer time horizons, has inspired three contemporary active inference agent algorithms. These address the problem of scaling the agents via the use of neural networks as function approximators. Following a discussion about the origins and mechanism of active inference and deep reinforcement learning, the three algorithms are studied. These draw inspiration from model-free and model-based deep reinforcement learning, with the third algorithm taking a hybridised approach.

*This report serves as the written project submission for a Bachelor of Science Honours degree at the University of Cape Town. The report seeks to study contemporary research at the intersection of active inference and reinforcement learning.

Contents

Notation

The notation below, follows [1]. We use this notation throughout all sections.

t, τ	Time index
s, s_t, s'	States of the environment
o, o_t	Observations
a, a_t	Actions
π, π'	A policy (sequence of actions, $a_1, a_2, a_3, \dots$)
π^*	The optimal policy
p	Known or unknown fixed probability density
q	An approximate probability density (variational density)
σ	Softmax function
D_{KL}	Kullback-Leibler divergence
$\mathcal{F}$	Variational free energy (VFE)
$\mathcal{G}$	Expected free energy (EFE)
$\mathcal{S}, \mathcal{R}, \mathcal{A}, \mathcal{O}$	Sets of states, rewards, actions and observations respectively.
r	Rewards
$\mathbb{E}$	Expectation function (mean)
C	Cumulative reward
γ	Discount factor hyperparameter (used for rewards)
V, Q or V_π, Q_π	State value function and state-action value function, respectively.
V^*, Q^*	Optimal state and state-action value functions.
$\mathcal{L}$	Objective (loss) function
$\tilde{s}, \tilde{a}, \tilde{r}$	Trajectories (for states, actions and rewards)
θ, ϕ, Ψ, ξ	Parameters (of neural networks)
$\tilde{p}$	Prior preference distribution (of the agent)
ϵ	Random noise term
α	Step-size hyperparameter
Ω	Binary variable representing optimality of a trajectory.
$\mathcal{H}$	Entropy function
$\tilde{\mathcal{F}}$	Free energy of expected future trajectories

1 Introduction

"How can the events in space and time which take place within the spatial boundary of a living organism be accounted for by physics and chemistry?" - Erwin Schrodinger [2]

Active inference [3] is a normative framework for a decision-making agent to generate behaviour. The framework is explicitly derived from the free energy principle which presents a global theory of self-organisation in contemporary theoretical neuroscience [4]. The principle asks the fundamental question: can one characterise the necessary behaviour of any self-organising non-equilibrium system that maintains a statistical separation from its environment? Applying the principle means that one can characterise such a system as performing approximate Bayesian inference. The active inference agent's behaviour can be thought to consist of three belief-updating schemes. These correspond to perception, action and learning which can all be interpreted as forms of inference for the active inference agent. The agent unites these in its imperative to minimise a unified objective function, the variational free energy [5]. This provides the means through which the agent performs approximate Bayesian inference. The central tenet of the active inference agent is, precisely, the minimisation of variational free energy.

Embedded in the minimisation of this objective function are the active inference agent's goals. The goals are encoded in a prior distribution over future states or observations [6]. When the agent minimises its objective function, it implicitly matches its predictive distribution over states of the environment to its distribution that is encoded with its preferred goal states. This embeds a goal-seeking behaviour in the agent. At the same time, the minimisation of free energy furnishes the agent with an exploratory drive. This facilitates the agent in gaining information to resolve uncertainty about its environment. Thus, the active inference framework addresses the problem of decision-making in an uncertain environment in order to achieve some goal. Moreover, the agent's goal-seeking and information-seeking behaviour provide a means for the agent to address the exploration-exploitation dilemma [7, 8].

Should an agent exploit its current knowledge to address its goals or, by doing so, is it sacrificing other unknown optimal action sequences that can better achieve these goals? This is the problem of the trade-off between exploration and exploitation [7, 8]. The trade-off arises since in order to obtain improved estimates about the behaviour of its environment, the agent may need to explore regions of its state-space away from its local optimum and in search of a global optimum for its choice of actions. However, the opportunity cost associated with ignoring the local optimum may pose a risk which is greater than the benefit obtained by exploring new territory [1]. Thus the problem of exploration poses a trade-off with exploitation. This problem is important for decision-making agents in uncertain environments. This encompasses reinforcement learning agents.

The reinforcement learning problem is one of how an agent can optimally select actions in an environment so as to maximise a numerical reward signal [8]. The agent must optimise its sequence of actions whilst finding itself in a changing landscape whereby its actions affect the state of its environment. It therefore attempts *trial-and-error* learning with the *reinforcement* learning aspect coming from the feedback provided by the numerical reward signal. These are the two key characteristics of reinforcement learning [8].

The usual formalisation of the reinforcement learning problem is in terms of the agent-environment interface and a Markov decision process (or partially observable Markov decision process). Here, the transition probabilities (between states of the environment) and the reward signal are often unknown to the agent. One common solution technique for addressing the problem of how best to optimally select actions so as to maximise cumulative reward, is to estimate state value functions. These attribute learned values to states of the environment based on the experience of the agent. The agent can then select the actions which maximise the value of the environmental states that it visits. Alternatively, policy functions, which are maps from states to actions, can be learned directly via the agent's experiences. These methods frequently employ approximations of dynamic programming to solve the action optimisation problem.

Classical reinforcement learning methods were often applied to small discrete state-spaces, where the value functions or policies were stored in matrices or vectors [8]. Contemporary reinforcement learning methods [9, 10, 11, 12], therefore, scale up the classical (tabular) methods through the use of powerful function approximators such as deep neural networks. These can be used to approximate the state value functions or policy functions to achieve better performance in larger state-action spaces. The experience of the agent's state, reward and action trajectories is used to provide the data necessary to implement such a supervised learning approach.

Similarly to classical reinforcement learning, a problem that has limited past active inference agents, is that of scaling [1, 13, 14]. Past approaches have been computationally limited by the need to evaluate all or many possible sequences of actions in small state-spaces [1, 6, 15, 16]. Hence, the primary premise underpinning this study is that active inference agents can benefit from the insights of deep reinforcement learning agents. Specifically we ask the question: how can methods from deep reinforcement learning be used to scale up active inference agents from tabular approaches, which address small state-action spaces, to deep-learning approaches, that use neural networks as function approximators to address environments with large state-action spaces?[1] This will be addressed via the study of three contemporary active inference algorithms [13, 14, 17].

1.1 Overview and Scope

To address the question of how to use deep reinforcement learning methods for scaling active inference agents, we begin by asking the question: what is active inference? Since the free energy principle of theoretical neuroscience provides the basis for the derivation of active inference, we discuss this in brief detail. We also provide some examples of active inference's use cases in computational psychiatry [18] and neuroscience [19].

Having justified our interest in active inference, we turn to a discussion of its mathematical formulation. First, it is necessary to establish the variational inference framework as a means of approximating Bayesian inference with an optimisation problem. Next follows the formulation of active inference as a partially observable Markov decision process, in which the agent performs perception, action and learning via the unified imperative to minimise variational free energy. A decomposition of the agent's variational free energy objective function provides a means of interpretation here. This reveals a Bayesian process for inference and learning [15].

A second objective function, called expected free energy, assumes that the agent minimises the variational free energy objective over time. This objective, which is necessary for action selection, is also described via decomposition for interpretation. This reveals that the agent's minimising of free energy embeds a reward-seeking imperative along with an imperative to gain information, explicitly to resolve uncertainty [15]. Lastly, since action selection is based on evaluating all possible trajectories of expected free energy into the future, a problem of scaling active inference beyond small state-spaces with short time horizons arises [1].

To address the problem of scaling using deep reinforcement learning methods, it is essential to address the questions of what it is we mean by the reinforcement learning problem and what some contemporary solution methods are. This is addressed by defining the agent-environment interface and a Markov decision process [8]. To establish a general solution framework for RL agents (to find their optimal policy), we discuss the Bellman equations and generalised policy iteration. This leads to the study of deep RL solution algorithms in both the model-free and, model-based settings.

[1] Conversely (and secondarily to the question of scaling active inference) the question of whether RL approaches can benefit from the directed intrinsic exploratory drive which furnishes the deep active inference objective function is touched on. However, the insights discussed here do not comprise the focus of the paper and are therefore incomplete (as compared to those of the problem of scaling).

With the basic formulation of reinforcement learning in mind and, with some solution algorithms in hand, we turn to framing the RL problem as a problem of inference [20]. One particular framing, in this regard, leads to *maximum-entropy RL* which furnishes the RL agent's objective function with an exploratory drive. This leads to a discussion of the exploration-exploitation dilemma where we discuss the benefit of the active inference agent's exploratory drive as a means of *directed* exploration which, specifically, provides an efficiency benefit [1].

Having discussed a framework for deep reinforcement learning and for active inference, we next discuss a means of reconciling these to provide the benefits of the scalability of deep reinforcement learning algorithms to active inference agents. We study three algorithms that achieve this. The first is inspired by a model-free RL approach [13] and the second by a model-based approach [14]. Lastly, after a discussion about the differences between the two approaches in terms of iterative and amortised inference [21], we address a hybridised approach [17].

More specifically, the model-free RL paradigm inspires an actor-critic algorithm. This is called *deep active inference* [13]. This algorithm uses neural networks to approximate the key distributions which appear in the variational free energy objective function of the active inference agent. One of these distributions is the policy network (the actor). The expected free energy functional provides an RL-inspired value function (the critic) to evaluate trajectories via a bootstrapped expression. Secondly, the model-based active inference algorithm provides a new objective function. This is called the *free energy of expected future* [14]. This algorithm makes use of model-based planning to evaluate trajectories and derive an optimal policy distribution. This distribution is a softmax distribution which assigns greater probability to policies with a lower measure of free energy. Lastly, upon discussion of a new taxonomy for RL algorithms, one that involves iterative and amortised inference [21], we turn to a hybridised approach between the model-free and model-based algorithms. This hybrid uses a model-free policy as the starting point for a model-based planning algorithm [17]. This combines the sample efficiency of model-based methods with the asymptotic efficiency of model-free methods. The study of these three algorithms comprises the means through which we address the question: how can active inference methods be scaled?

As our focus lies in scaling active inference agents to be able to tackle larger state-action spaces using deep reinforcement learning methods, much about the use of active inference in contemporary neuroscience is omitted.[2] However, within theoretical neuroscience, the advent of the free energy principle remains an interesting area for future study. In the area of machine learning the benefits of the active inference objective function, to provide a directed intrinsic exploratory drive, are mentioned. Testing the empirical performance of this drive on typical benchmark tasks [13, 14, 17], such as MountainCar [23], provide an entry point for future study here. In this work however, we maintain our focus on the problem of scaling active inference methods using deep reinforcement learning. Since both active inference and reinforcement learning agents can be seen to address the same problem (decision-making in uncertain environments, to achieve some goal), this provides the lens through which we view the problem of scaling.

2 What is Active Inference?

Active inference [3] is a normative framework for agent-based decision-making that generates behaviour based on minimising a functional called free energy [1, 6, 15]. The active inference agent's objective function groups perception, action and learning under one roof. Perception refers to inference about external (often hidden) states, using information from sensory observations of the world. Action refers to decision-making and planning. This can be viewed as inference as to the agent's policy, its (optimal) choice of sequences of actions in its environment. Learning can be seen as inference about the parameters of a generative model. A generative model is a joint probability model which depicts the relationship between how states

[2] Importantly, an interest in the active inference framework from the theoretical neuroscience community stems from the biological plausibility [15, 22] of the active inference process theory. However, this does not form the focus of this work.

of the world generate observations.[3] Inference about the agent's generative model parameters refers to the improvement of the agent's world model given new data.[4] Action, perception and learning (in the active inference agent-based decision-making framework) are, therefore, all inference schemes.

The primary imperative of the active inference agent is the minimisation of its variational free energy functional. This is a process of approximate Bayesian inference which turns the problem of inference into an optimisation problem. Through the imperative to minimise its free energy, the agent pools action, perception and learning into one unified approximate Bayesian inference framework for decision-making. The variational free energy objective function can be decomposed into terms representing accuracy and complexity [1, 15]. As we will show, when the agent seeks to minimise variational free energy, it looks to improve the accuracy of its world model, given new observations. However, the complexity term ensures that the agent's prior beliefs are taken into account in the construction of its new world model. This relates to the agent's desire to achieve its goals, which are encoded in a prior preference distribution.

Building an active inference agent consists of three processes [6, 15]: i) equipping the agent with a (generative) model of its environment, ii) training the agent by fitting it to observations via minimising variational free energy (and thereby performing approximate Bayesian inference) and, iii) having the agent select actions and make decisions that minimise its expected free energy. Importantly, the expected free energy functional is a quantity that encodes the agent's prior preferences or goals.[5] As the expected free energy functional can be decomposed into terms representing goal-seeking and information-seeking behaviour and, since it is used to select the agent's actions, the agent's action selection process is therefore a process of goal-seeking and information-seeking [1, 15]. This is important in the context of the exploration-exploitation dilemma [1].[6]

Exploration, for the active inference agent, refers to the agent seeking new observational data from its environment that will, specifically, update its world model to reduce uncertainty. Exploratory behaviour for the active inference agent is therefore *directed* since the agent specifically seeks to visit states that reduce uncertainty in its model. This contrasts to methods of exploration that involve random exploration of the state-space [1]. Since random exploration may result in exploring areas of the state-space that are not useful to achieving the agent's goals this can be viewed as more wasteful than the *directed* exploratory behaviour of the active inference agent. This identifies a crucial benefit of the active inference framework's behaviour, as compared to other methods of inducing exploratory behaviour in decision-making agents in uncertain environments [1].

The exploratory method, above, provides one characteristic of the active inference agent.[7] Another characteristic of the active inference agent is its exploitative behaviour. The active inference agent's model implicitly expects the agent's goals to be achieved. The model treats the agents preferences as the expected outcome of the agent's actions. The agent's exploitatory decision-making behaviour thus seeks to match its predictive distribution about the environment to its distribution of preferences. This process can be described as the agent conducting *self-evidencing* [1, 15].

Self-evidencing can be viewed as a homeostatic drive to remain in states of prior preference [24]. This characterises the process of minimising variational free energy to be the same process which preserves life from the tendency of disorder (which appears via the second law of thermodynamics) [25]. The minimal thermodynamic conditions required to sustain life, as articulated by the free energy principle, offer a means of addressing the *Hard Problem of Consciousness* in an application of free energy principle [19]. The agent's self-evidencing characteristic relates to the origin of the active inference framework, from the free energy principle. In fact active inference is explicitly derived from the free energy principle [3]. We will briefly

[3] For instance, the generative model $p(o, s) = p(o|s)p(s)$ is a joint probability density over states s and observations o, where the states generate the observations, as seen in the conditional probability statement $p(o|s)$.

[4] Learning generally happens over longer timescales than perceptual inference.

[5] The expected free energy can also encode the agent's preferences to receive greater amounts of reward, thus providing a one-to-one correspondence between active inference and traditional reinforcement learning agents.

[6] Later we will see this dilemma arises in the context of the reinforcement learning problem, in section 3.4.

[7] ... although not necessarily unique.

review the free energy principle to establish a story of these origins.

In the context of the free energy principle, the active inference framework has been used to simulate intelligent behaviour in neuroscience. For instance, active inference has been used for modelling planning and navigation [26], eye movements [27] and, for simulating behavioural anomalies analogous to psychiatric disorders, an approach called computational psychiatry [18]. In fact, active inference offers biological plausibility [22].

Hence, active inference modelling offers a useful framework in the context of neuroscience. However, the focus on the neuroscientific aspects of active inference do not form the focus of this paper. Instead we frame our problem as one of machine-learning and, specifically, the reinforcement learning problem and, exploration-exploitation dilemma [7, 8].

Active inference agents, similarly to reinforcement learning agents, can be applied to general planning problems for decision-making agents in changing environments. In fact, active inference and reinforcement learning agents address the same problem. This is the problem of how an agent can select an optimal policy in an uncertain environment so as to achieve its goals. One novelty of the active inference agent approach in this regard is from its, aforementioned, *directed* exploratory behaviour which is explicitly derived from its imperative to minimise expected free energy, as mentioned. This differs from other approaches to exploration which may be less efficient and more ad-hoc [1].

Motivated by this particular exploratory benefit, we endeavour to address the limitations of some early implementations of active inference agents which were often limited to small discrete state-spaces [1, 6, 15, 16].[8] This leads to a review of the tools of deep reinforcement learning which guide attempts to *scale* active inference to larger state-action spaces [1, 13, 14, 17].

2.1 The Origins of Active Inference: The Free Energy Principle

Active inference is a process theory that is explicitly derived from the free energy principle [3]. Although not the focus of this paper, it is worth mentioning the concepts of the free energy principle to establish the foundations of active inference. The following section provides a non-detailed, lightning introduction to the free energy principle.[9] However, this section can be read with the knowledge that a later section will provide a framework for active inference agent modelling that does not depend on the free energy principle. Hence this section provides a non-rigorous, light primer to the principle form which active inference agents are derived.

The free energy principle [4] describes a self-organising system (for example an active inference agent) that maintains itself at a non-equilibrium steady-state (the agent's preference distribution) and, whose states are governed by the statistical independence structure of a *Markov Blanket* [29]. The principle says that the internal states of such a system can be seen to perform inference on the external states (the agent holds an internal model of its environment) using approximate Bayesian inference and, that this is the process through which the system is able to resist the second law of thermodynamics, resist a natural tendency to disorder and, maintain its non-equilibrium steady-state [25] (via the agent taking actions to maintain its preferred states according to its internal model).[10]

The Markov Blanket [25] involves a partition of the states of the system into external states, s, internal states, μ and blanket states b. If the states of the system are represented by x then $x = \{s, \mu, b\}$. The blanket states separate the internal and external states via the statistical independence structure below. The Markov Blanket is defined by the following property of statistical independence, that the internal states are independent from the external states, given the blanket states [1]:

[8] Although the original application was in continuous time, state and action spaces [3].
[9] For greater mathematical detail see *A free energy principle for a particular physics* by Karl Friston [28].
[10] This is a homeostatic process.

$$p(s, \mu, b) = p(s|b)p(\mu|b)p(b) \tag{1}$$

The blanket states can be partitioned further into sensory states, o, and active states, a, such that $b = \{o, a\}$. Due to this separation, the Markov Blanket defines a causal loop of the states' influence on one another [28]. The external states can influence the sensory states (the agent can receive an observation from the environment). The sensory states can influence the internal states (the agent updates its model to take into account the new observation). The internal states can influence the active states to influence the external states (the agent can take actions on the environment to achieve its goals, which are encoded in its internal model). This can be seen in figure 1.

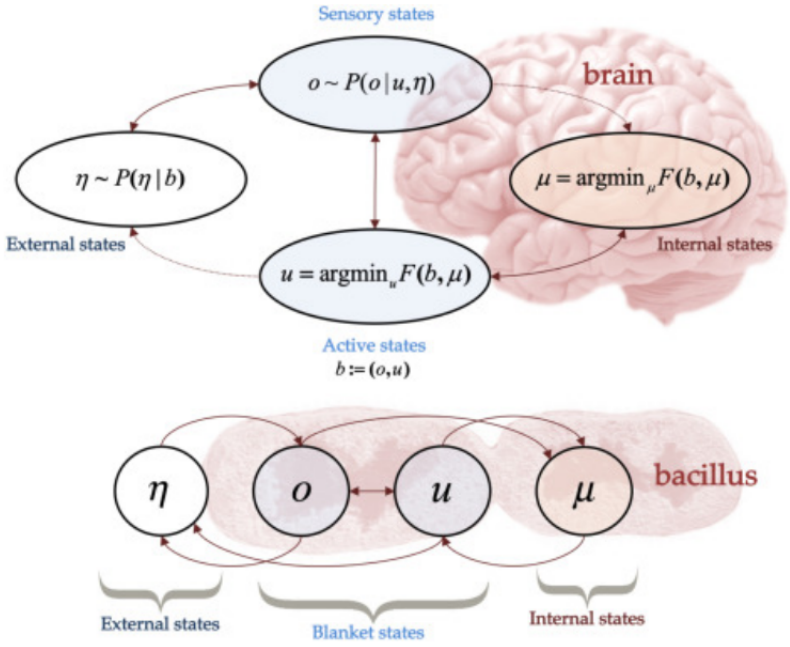

Figure 1: A Markov Blanket loop. The agent's internal states, μ, are separated from the external states, η (used here instead of s), via a Markov Blanket which consists of sensory states, o, and active states, u (used here instead of a). In this causal loop, the external (environmental) states can influence the sensory states which can influence the agent's internal states. The internal states can influence the active states which can influence the external states. The figure thus depicts the probability distribution of the external states, $p(\eta|b)$, as dependent on the blanket states. Similarly, the distribution for sensory states, $p(o|u, \eta)$, is dependent on the external and active states. The internal states minimise free energy (an imperative of the agent) to perform perception, which is approximate Bayesian inference about the external states. The active states minimise expected free energy to perform *active* inference. They therefore select actions that *self-evidence* meaning that they move the predictive distribution of the agent toward its distribution of preferences. This figure appears in [1] and is adapted from [28] where the brain and bacteria (bacillus) are referred to.

Active inference relates to the active states [1]. The account of active inference presented by the free energy principle says that, since active states can only influence external states, the active states bring the external states in line with the internal and blanket states' preferences about the external states. This process draws the internal states' beliefs toward the agent's preferences and is called *active inference*. It may be useful to examine the formal definition of active inference [24]. This defines the tuple $(\mathcal{E}, \mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{M}, p, q)$ underlying active inference. These are:

- $\mathcal{E}$: a sample of space from which random fluctuations $\epsilon \in \mathcal{E}$ are drawn.

- $\mathcal{S} : \mathcal{S} \times \mathcal{A} \times \mathcal{E} \to \mathbb{R}$: external states of the world that influence the agent's sensory states and depend on the agent's actions.
- $\mathcal{O} : \mathcal{S} \times \mathcal{A} \times \mathcal{E} \to \mathbb{R}$: the agent's sensory (observational) states that constitute a probabilistic mapping from active and external states.
- $\mathcal{A} : \mathcal{O} \times \mathcal{M} \times \mathcal{E} \to \mathbb{R}$: the agent's active states which depend on its sensory and internal states.
- $\mathcal{M} : \mathcal{M} \times \mathcal{O} \times \mathcal{E} \to \mathbb{R}$: the agent's internal states. These cause the agent's actions and depend on the agent's sensory states.
- $p(s, o, a, \mu|m)$: an ergodic density function over external states $\eta \in \mathcal{S}$, sensory states $o \in \mathcal{O}$, active states $a \in \mathcal{A}$ and internal states $\mu \in \mathcal{M}$ for a system denoted as m which is described as a stochastic differential equation in [24].
- $q(s|\mu)$: the variational density which is an arbitrary probability density function over external states, given internal states.

The free energy principle also refers to the agent's preference distribution [1]. This is described as the non-equilibrium steady-state $\tilde{p}$ that is held by the internal states of the system. The internal states of the system are thus assumed to *self-organise* toward this non-equilibrium steady-state [25]. Thus in the active inference framework the agent is treated as a system that self-organises to some non-equilibrium steady state, and that acts upon the world so that its predicted states match this target distribution. The target distribution, $\tilde{p}$, encodes the agent's goals [1, 6, 15].

While active inference is derived from the free energy principle it does not depend on it and can stand independently as a framework for agent-based decision-making, where the agent displays exploitative behaviour in search of achieving its goals and, exploratory behaviour in search of reducing uncertainty in its world model [1]. This is the treatment we adopt in this paper. This is important due to philosophical debate as to the applicability of the free energy principle . As a (relatively) new theory it remains of interest to explore the veracity of the assumptions needed to apply the principle. Namely, the assumption of a dynamical system that self-organises toward a non-equilibrium steady-state and, the assumption that this system holds the statistical independence structure of a Markov blanket, may limit the free energy principle's applicability especially with regard to the study of complex systems such as the brain.

Hence, the account of active inference presented here does not rely on the free energy principle, although analogy to it can be drawn. In light of this, we continue with our intended study which views the active inference framework as a framework for agent-based decision-making, in an uncertain environment, to achieve some goal. Interestingly, the active inference agent also holds the aforementioned *directed* exploratory drive which provides an additional motive. Next, it is necessary to establish the *variational inference* framework [5] and see how it serves to perform approximate Bayesian inference. This is crucial in uncovering the process through which the active inference agent selects actions, their imperative to minimise free energy.

2.2 Variational Inference

The framework of variational inference offers the basic mechanism for how active inference agents operate. Variational inference [5] offers an approximation technique to Bayesian inference. Bayesian inference looks to ascertain the posterior distribution, $p(s|o)$ from a prior distribution $p(s)$ and a likelihood function $p(o|s)$. This can be interpreted as inferring the states of the world, s, when presented with observations o. To do this one employs Bayes' theorem [30]:

$$p(s|o) = \frac{p(o|s)p(s)}{p(o)} \tag{2}$$

$p(o)$ represents the probability distribution of the data given the (implicit) model m, $p(o|m)$ (where m has been suppressed, as in [15]). $p(o|m)$ can therefore be interpreted as the model evidence whereby larger probabilities for $p(o|m)$ provide support for the model, m [15].

Exact Bayesian inference requires the calculation of $p(o)$, however this can often involve an intractable integral $p(o) = \int dsp(o|s)p(s)$.[11] To replace the intractable integral, variational inference is used [1, 15]. This performs an approximation of Bayesian inference by replacing the intractable integral with an optimisation problem.

The calculation of the posterior $p(s|o)$ is replaced with the approximation of the *variational density* $q(s|o;\theta)$, where θ are a set of parameters. The variational density is optimised to approximate the true posterior.

To optimise the variational density, variational inference minimises a measure of the distance between the variational density and the true posterior density. This measure is the Kullback-Leibler divergence and is measured in terms of the information theoretic units, nats.[12] The KL divergence between two probability densities, $p(s)$ and $q(s)$ is defined as:

$$D_{\mathrm{KL}}(q(s)\|p(s)) = \int dq(s) \ln\left(\frac{q(s)}{p(s)}\right) \tag{3}$$

The KL divergence has a minimum of 0 when the two probability densities are equivalent. Variational inference minimises the following KL divergence between the true posterior and the variational density, thus bringing the approximation closer to the true posterior. The optimal parameters of the variational distribution that bring it closest to the true posterior are given by:

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\ D_{\mathrm{KL}}[q(s|o;\theta)\|p(s|o)] \tag{4}$$

However, expression 4 poses the problem that it contains the intractable true posterior, which is yet to be estimated. Instead of calculating this, variational inference minimises a tractable bound on the KL divergence in expression 4. Variational inference calls this tractable bound *variational free energy* which is defined as [1]:

$$\begin{aligned} \mathcal{F}(o,\theta) &= D_{\mathrm{KL}}[q(s|o;\theta)\|p(s,o)] \\ &= D_{\mathrm{KL}}[q(s|o;\theta)\|p(s|o)] - \ln p(o) \\ &\geq D_{\mathrm{KL}}[q(s|o;\theta)\|p(s|o)] \end{aligned} \tag{5}$$

Thus the KL divergence between the approximate posterior, $q(s|o)$, and the true posterior, $p(s|o)$, is equal to the KL divergence between the approximate posterior and $p(s,o)$ plus the natural logarithm of the marginal probability of the observational data, $p(o)$. As mentioned, $p(s,o)$ is often called the *generative model* [15] of how hidden states s generate observational data o. Since the generative model is $p(s,o) = p(o|s)p(s)$ then if an (arbitrary) initial variational posterior $q(s|o)$ is specified, we have what we need to calculate the upper bound on the KL divergence between the true posterior and variational density. Since the upper bound is tight as seen in equation 5, then by minimising this tractable upper bound on the divergence between the posterior and approximate posterior, we improve upon our estimate of the true posterior and therefore accomplish approximate Bayesian inference. One can see from equation 5 that the distribution $q(s|o;\theta)$ which best minimises $\mathcal{F}(o;\theta)$ is the distribution $q(s|o;\theta)$ which best approximates the true posterior since when these are equal their KL divergence is zero (and since KL divergence is non-negative). Later we will see that one can accomplish variational inference using gradient descent on the variational free energy

[11] ... except for the case of a small set of tractable pairings of distributions.
[12] Nats are the natural logarithmic equivalent of bits [15].

[1, 13, 15]. Next we turn to studying a framework for understanding the active inference agent and for understanding how it unites the inferential processes of perception, learning and action when minimising variational free energy to achieve its goals.

2.3 Perception, Learning and Action

As previously mentioned, active inference is a normative theory unifying perception, action and learning under the single objective of minimising free energy. More specifically, the *active* component of the framework (the component of action selection and decision-making) chooses to minimise an expected free energy functional [1, 15]. As mentioned, this encodes goals in terms of a prior over future states (or observations). Here we provide an account of discrete state-space active inference and discuss the mechanism through which active inference agents operate. The active inference framework reviewed in this section will later be expanded, in order to study how active inference agents can be scaled using insights from deep reinforcement learning.

The active inference agent's problem of decision-making in an uncertain environment to achieve its goals can be formalised in terms of a partially observable Markov decision process (POMDP) [1, 6, 15]. Formally this is a tuple $(\mathcal{S}, \mathcal{A}, P, \mathcal{R}, \mathcal{O}, O)$ of states $\mathcal{S}$, actions $\mathcal{A}$, transition probabilities P, rewards $\mathcal{R}$, observations $\mathcal{O}$ and, observation probabilities O.[13] The POMDP means that the agent is: (i) subject to the Markov property whereby the relevant history of the system is contained in the system information from the previous time-step along, (ii) the agent is in a Markov decision process whereby there are several states of the system, the agent can make decisions which impact the state it finds itself in and, the states generate an amount of reward that the agent receives and, lastly, (iii) the agent cannot (necessarily) directly see the state of system but rather must infer this (hidden state) from the observations it receives. An example of a POMDP appears in figure 2.

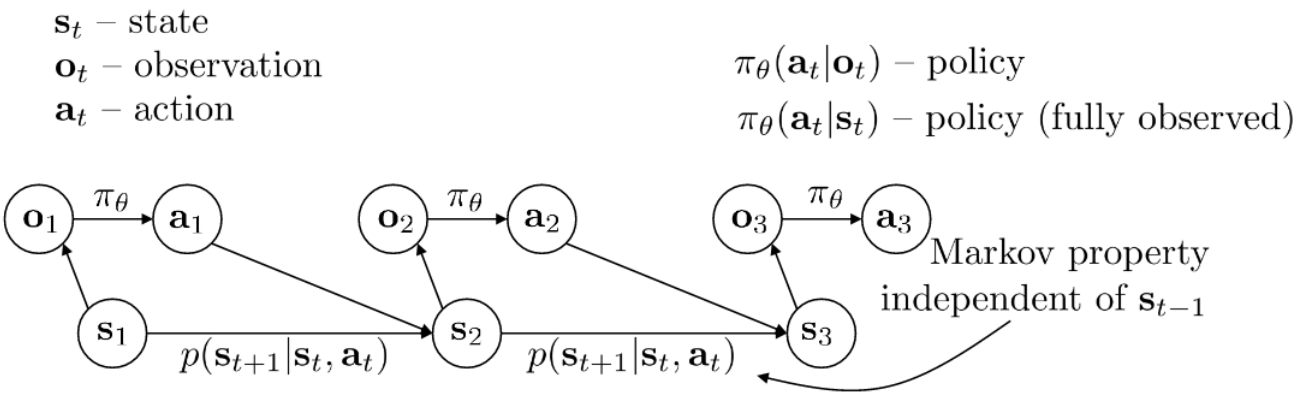

Figure 2: An example of a partially observable Markov decision process (POMDP). One can see from the graph that the states of the environment, s, generate the agent's observations, o. From the observations the agent can infer the state that it is in and use its policy to take an action, a. Note that the policies, $\pi_\theta(a_t|o_t)$, are policies of a reinforcement learning agent too. This is since the active inference agent and reinforcement learning agent tackle the same problem, how to make decisions in an uncertain environment so as to maximise their goals. Given the choice of action and true state of the environment, the POMDP has some transition probability $p(s_{t+1}|s_t, a_t)$. The image is from [31].

Given the POMDP the active inference agent can perform the following [1]:

[13] Interestingly, this bears similarity to the definition of the active inference tuple of the previous section. In fact, active inference agent's can be formalised as a POMDP [1, 6, 15].

- Perception: The agent conducts inference on the hidden states of the world, $s \in \mathcal{S}$, from the observations, $o \in \mathcal{O}$, that it obtains and which are generated by the hidden states.
- Learning: The agent's model may possess parameters θ which can be incrementally optimised over time thereby improving the model for inference.
- Action: The agent can make decisions to achieve a desired goal. Action selection consists of inferring an action $a \in \mathcal{A}$ or, inferring some sequence of actions called a policy, $\pi = \{a_0, a_1, ...\}$.

Similarly to variational inference, the central tenet of the active inference agent is the minimisation of variational free energy (VFE). As mentioned, this unites the three components of action, perception and learning under a single objective function. VFE can be redefined (from what appeared before in equation 5) to include the parameters, θ, in a Bayesian manner. This is whereby the parameters, θ, are included as random variables and therefore as part of the probability distribution and inference procedure. This allows for inferential reasoning about the uncertainty in the model parameters. One choice of definition for VFE that follows [1] (and includes states, observations and parameters) is:[14]

$$\mathcal{F}(o) = D_{\mathrm{KL}}[q(s, \theta|o) \| p(s, o, \theta)] \tag{6}$$

where the agent's variational posterior distribution is $q(s, \theta|o)$ and its generative model is $p(s, o, \theta)$. The implementation of the agent's perception, action and learning schemes involve the specification of the generative model and the variational distribution as these completely specify the model. Subsequently, to implement perception, action and learning the agent performs the following minimisations of variational free energy [1]:

- Perception:
$$\underset{q(s|o)}{\operatorname{argmin}}\ \mathcal{F}(o) \tag{7}$$
The agent's perception is an inference scheme that minimises VFE to obtain the optimal posterior for states given observations (and therefore for the perception of states given sensory observations).
- Learning:
$$\underset{q(\theta|s,o)}{\operatorname{argmin}}\ \mathcal{F}(o) \tag{8}$$
The agent's model learning involves the minimisation of VFE to obtain an optimal posterior for model parameters θ.
- Action:
$$\underset{a}{\operatorname{argmin}}\ \mathcal{F}(o(a)) \tag{9}$$
The agent selects actions that best minimise its VFE in the future. As this version of VFE is measured in terms of the posterior for states and θ, *given observations*, the agent will select actions that lead to it receiving observations that minimise VFE. This can be interpreted as the agent selecting actions that bring its predictive (variational posterior) distribution, q, closer to its internal model of preferences and beliefs, $\tilde{p}(s, o, \theta)$.

Note that perception refers to inference about the posterior for the hidden states, $q(s|o)$, whereas learning refers to the posterior for model parameters $q(\theta|s, o)$. Hence together these give the joint posterior $q(s, \theta|o) = q(s|o)q(\theta|s, o)$ which can be obtained from jointly minimising the VFE.

[14] Other definitions of free energy that include actions in the posterior will be used later in this paper. For now note that, in general, VFE refers to an upper bound on the KL divergence between the variational posterior and true posterior. Hence minimising VFE optimises the variational posterior to better approximate the true posterior.

Given this procedure we now know that specifying the generative model and variational distributions completely specifies the model underlying the active inference agent and, which gives it access to the processes of perception, action and learning. However, the above implementation of action (a minimisation of VFE) applies to continuous models [3] and was included to illustrate that the free energy minimisation process gives inference of all three of perception, action and learning. However, due to the dependence of VFE on observations which are dependent on actions, the action minimisation is non-trivial [1]. An alternative (and commonly used) approach used for discrete state-spaces [1, 6, 15, 16, 32] is the following: assume a functional form for the variational posterior over policies, $q(\pi)$. This is set equal to a softmax distribution over the negative *expected free energy*, $-\mathcal{G}(o, s)$, which is defined in the next subsection.

- Action (discrete):

$$q(\pi) = \sigma(-\mathcal{G}(o, s)) \tag{10}$$

where the softmax function, σ, for an input x, is defined as [15]:[15]

$$\sigma(\mathrm{x}) = \frac{\mathrm{e}^{\beta \mathrm{x_i}}}{\sum_{\mathrm{k}} \mathrm{e}^{\beta \mathrm{x_k}}} \tag{11}$$

The softmax function for a set of discrete inputs yields a categorical distribution. The function has a monotonic relationship with the input variables hence, a larger *negative* expected free energy value (for a particular policy) would yield a lower probability of that policy being executed by the agent [15]. As with VFE, the active inference agent seeks to minimise expected free energy. Hence, the optimal distribution over policies can be viewed as one which selects policies in proportion to the exponentiated (negative) expected free energy and hence one which (most frequently) selects policies that minimises expected free energy.

The framework of the active inference agent's decision-making thus encompasses two objective functions. First, the VFE objective function is used for the purposes of the perceptual inference and, learning schemes. Secondly, the expected free energy is used to evaluate potential policies. The following decomposition offers an interpretation of the minimisation of VFE [1, 15] (and the EFE objective is discussed in the next sub-section):

$$\mathcal{F}(o, \theta) = \underbrace{-\mathbb{E}_{q(s|o;\theta)}[\ln p(o|s)]}_{\text{Accuracy}} + \underbrace{D_{\mathrm{KL}}[q(s|o;\theta)\|p(s)]}_{\text{Complexity}} \tag{12}$$

The first term of the decomposition interprets the minimisation of VFE as the maximisation of model accuracy. The accuracy term poses the problem of inference as one of maximising the likelihood of expected observations and, therefore, maximising the accuracy of the model in terms of new observations. In contrast the complexity term acts as a regularisation term which keeps the variational distribution closer to the prior distribution of states. This ensures that the variational inference procedure is, not only a maximum likelihood procedure but, is a procedure of Bayesian inference due to the incorporation of the prior. Next, we turn to a discussion of the second objective function, expected free energy. This objective introduces a temporal aspect to the active inference agent's decision-making.

[15] Here, β is a parameter that controls the spread of the softmax distribution. This is a hyperparameter that represents the precision of the estimation of EFE [15].

2.4 Expected Free Energy

For active inference agents, action selection inverts the usual question asked when choosing actions [1]. Instead of asking which action sequence will deliver its preferences, the agent assumes that it will achieve its preferences and asks the question: "given the assumption that I will achieve my preferences, which action sequence will I most likely pursue?"[16] The answer to this question is the policy $\pi = \{a_0, a_1, ...\}$ (sequence of actions) that best minimises the expected free energy (EFE).

The definition of EFE can be understood similarly to VFE (expression 6), as EFE is defined as the *expected* difference between the natural logarithm of the variational posterior and, the natural logarithm of a biased generative model under a specific policy. The biased generative model encodes the prior preference distribution (defined as $\tilde{p}$) which can encode preferences for observations, o, thus encoding goals which the agent assumes will be achieved. The prior preference distribution encoded for observations is $\tilde{p} = \tilde{p}(o)$. This gives the biased generative model, $\tilde{p}(o, s) = p(s|o)\tilde{p}(o)$ [1]. The expectation in EFE, is taken with respect to the approximate joint variational posterior for observations and hidden states at that time. Therefore, EFE encodes a distance metric, one which measures the expected distance of the predictive variational posterior distribution, q, from the prior preference distribution p. Hence the active inference agent, when minimising this distance measure (and hence minimising EFE), acts to bring its variational predictive distribution in line with its prior distribution of preferences. Thus defining EFE as per [1, 15] (and noting that since observations have not yet occurred, they enter the expression as random variables) gives that for policy π, the EFE is:

$$\begin{aligned}
\mathcal{G}_\pi(o, s) &= \mathrm{E}_{q(o,s|\pi)}[\ln q(s|\pi) - \ln \tilde{p}(o, s|\pi)] \\
&= \mathbb{E}_{q(o,s|\pi)}\left[\ln q\left(s|\pi\right) - \ln q\left(s|o, \pi\right) - \ln \tilde{p}\left(o_t\right)\right] \\
&= \underbrace{\mathrm{D}_{\mathrm{KL}}(q(o|\pi)\|\tilde{p}(o))}_{\text{Risk}} + \underbrace{\mathrm{E}_{q(s,o|\pi)}[\ln p(o|s)]}_{\text{Ambiguity}} \\
&= -\underbrace{\mathbb{E}_{q(o,s|\pi)}[\ln \tilde{p}(o)]}_{\text{Extrinsic Value}} - \underbrace{\mathbb{E}_{q(o|\pi)} D_{\mathrm{KL}}[q(s|o, \pi)\|q(s|\pi)]}_{\text{Intrinsic Value}} + \underbrace{\mathbb{E}_{q(o|\pi)} D_{\mathrm{KL}}[q(s|o, \pi)\|p(s|o, \pi)]}_{\text{Posterior Divergence}}
\end{aligned} \tag{13}$$

Since action selection (seen in expression 10) relies on the minimisation of EFE it is important to examine what this minimisation means for the agent. The first decomposition of EFE, above, provides two terms called risk and ambiguity. Minimising EFE minimises risk which can be interpreted as minimising the KL divergence between the agent's predictive posterior distribution for observations, $q(o|\pi)$ and the agent's prior preferences, $\tilde{p}(o)$. This drives the agent to seek actions which achieve its preferences for certain observations. This may be viewed as the agent's reward-seeking and exploitative behaviour (in the context of the agent's goals). Minimising EFE simultaneously minimises ambiguity, this can be interpreted as the agent avoiding expected observations (given expected states under a policy) which it views as being highly uncertainty [1].

For the second decomposition, a common interpretation of the first term, *extrinsic value* [32], is that of the agent maximising its expected reward (expected under the variational predictive distribution) [1]. Hence minimising EFE maximises the extrinsic value gained by the agent. The *intrinsic value* term can be interpreted as information gain [1]. Here, the minimisation of EFE encourages the agent to maximise the divergence between the variational prior and variational posterior over states. This term provides the specific insight that the agent is encouraged to visit states that will lead to an maximal increase in

[16] In terms of the free energy principle, this is where the pre-condition of a non-equilibrium steady state is applied. The free energy principle is applied to systems that self-organise toward a non-equilibrium steady-state. The active inference agent's prior preference distribution, $\tilde{p}$, is treated as the non-equilibrium steady-state to which the system self-organises via action. This affords the construction of the dynamics necessary to apply the free energy principle, namely that there exists a self-organising dynamical system with a Markov Blanket which approaches a non-equilibrium steady state. The principle then states that this system conducts approximate Bayesian inference via the minimisation of variational free energy.

information. The intrinsic value term demonstrates the benefit of the active inference agent, that it holds a *directed* intrinsic exploratory motivation. Later, we will contrast this with random exploratory methods which achieve exploration in a less efficient manner, as they visit states on a random basis, and not to increase information in a maximal and efficient way.[17]

With EFE in hand one can see that the two objective functions of the active inference agent, VFE and EFE work in tandem. Specifically, minimising VFE serves to allow the agent to learn an *accurate* world model via perception and learning, thereby allowing it to infer the hidden states of the world. Minimising EFE provides the agent with a policy by which to take actions. The policy offers goal-seeking and information-seeking behaviour to the agent. However, the quality of inference about policies depends upon the quality of the world model, resulting from minimising VFE [1]. This is because, in order to predict which action sequences fair better relative to others, the agent is required to make accurate multi-step predictions of the consequences of its actions. And since the expectation of EFE is taken with respect to the learned model, these multi-step predictions are sensitive to the quality of the model.

Refer back to the expression for action, expression 10, and recall that the softmax function will give the highest probability to the lowest EFE value. Now note, a key assumption of the active inference framework is the assumption of ergodicity of the system [1]. This means that the temporal average coincides with the state average for the EFE. Hence expression 10 effectively says that the optimal policy is a softmax distribution over the path-integrals of the EFE through time, with the most likely action sequence being that with the lowest EFE. Due to the ergodicity assumption, the path integral can be calculated as:

$$\mathcal{G}_\pi(o, s) = \sum_t \mathcal{G}_\pi(o_t, s_t) \tag{14}$$

Thus in small, discrete state-spaces the path integral can be computed exactly for all possible trajectories of actions. This affords the agent the ability to plan with its model, via the evaluation of all possible trajectories. Given trajectories with an evaluated EFE, the agent then possesses a policy via the softmax distribution in expression 10. This can be seen in the following discrete state-action space example.

A Discrete Active Inference Example

This section presents an example of a discrete state-action space active inference agent. The example is based on [33].[18] Suppose we have a simple agent whose environment is defined as having two states corresponding to having a full or empty stomach, $\mathcal{S} = \{\text{full, empty}\}$. The corresponding observation space denotes whether the agent feels fed or hungry, $\mathcal{O} = \{\text{fed, hungry}\}$. Suppose that the agent holds a prior preference for feeling fed and not hungry. This means that one possible encoding for its prior preference distribution will assign:

$$\tilde{p}(o) = \{\tilde{p}(\text{fed}), \tilde{p}(\text{hungry})\} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{15}$$

The agent's action space consists of the actions to eat or, to do nothing $\mathcal{A} = \{\text{eating, nothing}\}$. Suppose that the agent has a known likelihood matrix, $A := p(o|s)$, which gives the probabilities of being hungry or fed, given that the agent's stomach is full or empty. Assume, also, that there is a known transition probability matrix $B := p(s_{t+1}|s_t, a_t)$. This is defined for the action of eating and, for when the agent does nothing. This provides two transition matrices B_1 and B_2 which are 2×2 matrices that provide

[17] The last term in the expression for EFE, posterior divergence refers to the agent's minimisation of the error between its variational posterior distribution and the true posterior distribution. This term is often assumed to be small.

[18] A similar example, involving perceptual inference only, appears in [34]. Here, the author suggest that this inference procedure holds biological plausibility, an insight which is also drawn from [22].

the transition probabilities between the agent's states of the environment. Figure 3 provides the known information of the agent.

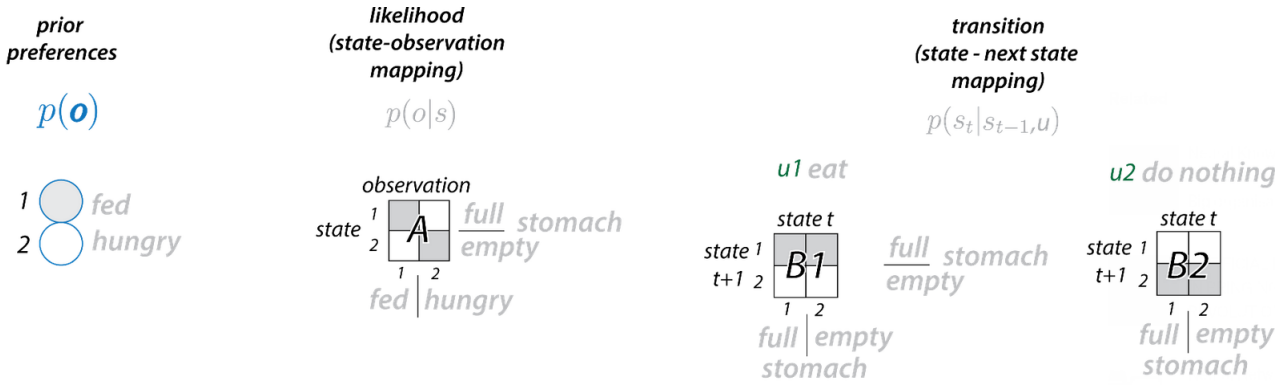

Figure 3: An example of a hungry active inference agent's prior known information, from [33].

Next, the agent is supplied with the observation that it is hungry: $o_1 =$ hungry. This decision-making agent wants to conduct an optimal sequence of two actions that will lead to it achieving its prior preference distribution, $\tilde{p}(o)$. As the action space only consists of two actions, there are only four possible policies. The agent will evaluate all four of these, in terms of their expected free energy. The agent's variational policy posterior can then be calculated as, $q(\pi) = \sigma(-\mathcal{G})$ (see expression 10).

The agent is given some prior distribution, $p(s_1)$, for the states that generated the initial, hungry observation. Armed with this knowledge, knowledge of the likelihood function and knowledge of the transition model, the agent evaluates the posterior for states given its observation of hunger, $q(s_1|o = o_1)$. This can be computed via the minimisation of variational free energy which, as per equation 12, maximises the accuracy of the state posterior with respect to observations, whilst incorporating the state prior. In terms of our food example and from the agent's perspective, the accuracy term updates its variational probability model by maximising the likelihood of its observation of being hungry. However, the complexity term constrains this update such that the posterior distribution for states s given that the agent is hungry does not diverge too far from the agent's prior $p(s)$. In this instance however, as we have access to the prior and likelihood matrices, this step can be computed using exact Bayesian inference.

Using the posterior, $q(s_1|o_1)$, and transition model, $p(s_2|s_1, a_1)$, the agent can derive the posterior for states given actions, $q(s_2|a_1)$, for all possible actions.[19] Given $q(s_2|a_1)$ and the known likelihood model $p(o_2|s_2)$, the agent can derive a posterior for observations $q(o_2|a_1)$.[20] These four terms, $q(o_2|a_1)$, $q(s_2|a_1)$, $p(o|s)$ and $\tilde{p}(o)$, allow for an evaluation of EFE for all possible choices of action. This is evaluated using a form of EFE from [15]:[21]

$$\begin{aligned}\mathcal{G}_{a_1}(o_2, s_2) &= \mathrm{E}_{q(o_2,s_2|a_1)}[\ln q(s_2|a_1) - \ln \tilde{p}(o_2, s_2|a_1)] \\ &= \underbrace{D_{KL}(q(o_2|a_1)\|\tilde{p}(o_2))}_{\text{Risk}} + \underbrace{\mathbb{E}_{q(s_2|a_1)}[\mathcal{H}[p(o_2|s_2)]]}_{\text{Ambiguity}} \\ &= D_{KL}(q(o_2|a_1)\|\tilde{p}(o_2)) + \sum_{\mathcal{S}} q(s_2|a_1)\mathcal{H}[p(o_2|s_2)]\end{aligned} \tag{16}$$

The resulting values of EFE can be summed together with the values of EFE for the next time-step (thus applying the ergodicity assumption of active inference). These will be computed using the current-time posterior distributions, $q(o_3|\pi)$, $q(s_3|\pi)$ along with the known distributions $p(o_3|s_3)$ and $\tilde{p}(o_3)$ and, given choices of action a_2 (and where $\pi = \{a_1, a_2\}$). Given the four values of EFE, each resulting from a two-step

[19] ... by averaging over possible states s_1 using the posterior $q(s_1|o_1)$. This is tractable due to the small discrete state-space.
[20] ... by averaging over possible states s_2 using the posterior $q(s_2|a_1)$.
[21] Here, $\mathcal{H}$ denotes the entropy of the argument.

summation, the agent can calculate its variational action posterior as $q(\pi) = \sigma(-\mathcal{G})$. This results in policies with smaller EFE having a higher probability of being executed by the agent. Thus, policies for which the observation posterior, $q(o|a)$, most closely matches the preferences of the agent, $\tilde{p}(o)$, are more likely to be executed. This corresponds to the minimisation of the risk term in equation 16). Furthermore, the (more frequent) selection of policies for which EFE is minimal, results in a (more frequent) selection of policies which minimise ambiguity and thus the agent's expected uncertainty about future observations.

We can also interpret the agent's behaviour in terms of the extrinsic value and intrinsic value decomposition from equation 13. This affords us the view that when choosing to minimise EFE, the agent selects actions that will maximise its extrinsic value term which is expressed in terms of its prior preference distribution. This implies that the agent, when minimising EFE, will assign more weight in its variational policy posterior to the action corresponding to its preference for feeling fed, as this yields $-\ln(1) = 0$ which is less than the infinite postive value corresponding to its preference for feeling hungry. This demonstrates that the agent holds an infinitely greater preference for feeling fed than hungry. Moreover, the selection of actions that minimises EFE yields a selection of actions that maximises the KL divergence between the posterior and prior for states. This corresponds to a selection of actions for which the agent gains the most information. In this example since the preferences of the agent are so stark, the extrinsic value term dominates the intrinsic value term in the minimisation of EFE. However, if for instance, the agent did not hold a preference for feeling either fed or hungry then the intrinsic value term could guide the agent to select actions such that, given a new observation, the posterior for states would maximally diverge from a prior for states (where the prior corresponds to an action choice). This affords the agent an intrinsic exploratory motive to explore its state-action space and, specifically, to explore states where uncertainty can be resolved and therefore, where the intrinsic value term is large.

Interestingly, the process of enumerating policies using EFE (with a time horizon of two steps) corresponds to *model-based planning* as the agent uses its model to enumerate trajectories and improve its policy. This process of enumerating all possible trajectories of actions does not scale well to larger state-action spaces and, to longer time horizons. This poses the problem of scaling for active inference agents. In fact, to evaluate all possible trajectories in the state-action space scales exponentially with the size of the state-action space [1]. Thankfully, the field of reinforcement learning has developed many algorithms to deal with this problem. Later we will approach the scaling problem by studying approaches inspired by model-free [13] and model-based [14] deep reinforcement learning. Next, we introduce the reinforcement learning problem.

3 Reinforcement Learning

The reinforcement learning (RL) problem is that of how an agent can optimally select actions in an uncertain environment so as to maximise a numerical reward signal [8]. RL can be viewed in contrast to the other subsets of machine learning (ML), supervised and unsupervised learning. While supervised learning presents an instructive learning method via labelled data and while unsupervised learning seeks to find implicit data structure in unlabelled data, RL can be considered an evaluative method. The RL agent's decisions are not classified as correct or not but instead are evaluated through the feedback provided by the reward which is received from the environment. The two key characteristics of the RL agent are firstly, that it engages in *trial-and-error* learning through its interactions with its environment and secondly, that it engages in *reinforcement* learning with the reinforcement aspect coming from the (often delayed) reward signal.

The agent find itself in a changing environment. It does not know the dynamics of this environment but must still attempt actions which affect the state of its environment and then assess which of these actions achieves the most cumulative reward (this is trial-and-error learning). The central tenet of the RL agent is, precisely, the maximisation of cumulative reward. The rewards from the environment provide the evaluative feedback, necessary so that the agent can improve upon its actions and resolve its uncertainty via *reinforcement* learning. As active inference agents and RL agent's both address the RL problem, we

will see later on how active inference agents can be posed as RL agents.[22] The usual formulation of the RL problem is in terms of Markov decision processes. This formulation is reviewed below.

Classical RL posed the problem of scaling, as solution approaches were applied to small discrete state-spaces over short time horizons [8]. Contemporary RL methods have addressed this issue. Approaches such as *policy gradient methods* [10], deep Q networks [9] and, actor-critic methods [10, 11] make use of deep neural networks as powerful approximators to scale classic RL techniques to achieve performance on complex tasks in larger state-spaces. For instance an RL agent has dominated the world champion at the complex game of Go [35], RL agents have seen success at playing the Atari games [9] and, RL agents have found success in the domain of robotics [36]. Thus the RL field has achieved a powerful general learning framework which has been used at scale, to successfully solve computationally complex tasks.

Due to the strong performance of deep reinforcement learning on complex tasks, we review these approaches as inspiration for addressing the scaling of active inference agents. The following section discusses the Markov decision process formulation of RL. Following [8], this gives a basic overview for a framework of solution methods called generalised policy iteration. Thereafter, we review some deep reinforcement learning algorithms in the model-free and model-based settings. This is followed by a framing of the reinforcement learning problem as one of inference [20]. Lastly a discussion about the exploration-exploitation dilemma justifies our particular interest in active inference agents in the context of the RL problem.

3.1 Tabular Reinforcement Learning

A simple RL algorithm in discrete time can be thought of as follows [8]. At each time-step the agent will observe the state of the environment and make a decision to take some action. This decision may be based on an arbitrary policy. The environment will respond by possibly changing its state and providing feedback to the agent through some reward. The agent can then use the information provided by this reward to inform its view on the action that it took and, to improve upon its policy. This is all motivated towards the goal of maximising the amount of cumulative reward received. With this basic idea of RL in mind we next study a mathematical formalism which will aid in our discussion of the RL learning algorithms.

Markov Decision Processes

The RL problem can be formalised in terms of a Markov decision process (MDP) [8]. Previously, a POMDP was described by figure 2. In contrast an MDP allows for observable states of the environment. An MDP appears in figure 4. Here, the agent chooses actions, using its policy which is a function $\pi : \mathcal{S} \to \mathcal{A}(s)$ from states to actions. The policy can be deterministic or stochastic, as we will see later. The figure illustrates the discrete time RL algorithm from the previous paragraph. The MDP formalism encapsulates the RL dynamics in terms of a stochastic process displaying the Markov property.

The Markov property[23] is used below where the joint probability distribution of a new state, s_t and reward, r_t, given the state-action pair, s_{t-1}, a_{t-1} from the previous time-step is defined. This applies for all $s_t \in \mathcal{S}, r_t \in \mathcal{R}$, and $a_t \in \mathcal{A}(s)$, $t \in 1, 2, 3, ...$ and $p : \mathcal{S} \times \mathcal{R} \times \mathcal{S} \times \mathcal{A} \to [0, 1]$. Following [8] this is:

$$p\left(s_t, r_t | s_{t-1}, a_{t-1}\right) \tag{17}$$

With this probability function in hand the state-transition probabilities are given by:

[22] Specifically, this involves either describing the active inference agent's prior preference distribution as preferring greater amounts of reward or, understanding the RL agent's desires in terms of general preferences instead of just for high cumulative reward.

[23] The Markov property says that information about a future variable can be expressed as well with information about variables at the previous time-step alone, as it can with information about those same variables across all past time-steps.

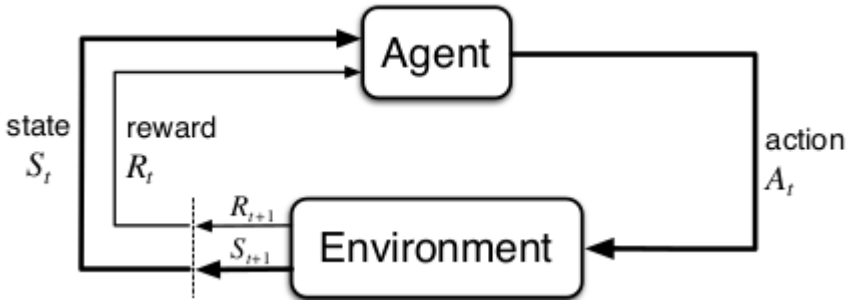

Figure 4: A Markov Decision Process (MDP) which describes the agent-environment interaction [8]. The agent, which finds itself in some state s_t, will select an action, a_t, which influences its environment. The agent will observe the new state of the environment, s_{t+1}, along with some reward, r, received from the environment. The agent must then select a new action using this information. The central tenet of the RL agent is to selects actions so as to maximise its amount of cumulative reward received from the environment.

$$p\left(s_t|s_{t-1}, a_{t-1}\right) = \sum_{r\in\mathcal{R}} p\left(s_t, r_t|s_{t-1}, a_{t-1}\right) \tag{18}$$

and the expected rewards for state-action pairs are:

$$\mathbb{E}\left[r_t|s_{t-1}, a_{t-1}\right] = \sum_{r\in\mathcal{R}} r \sum_{s_t\in S} p\left(s_t, r_t|s_{t-1}, a_{t-1}\right). \tag{19}$$

Thirdly, the expected reward for the state-action-next action triple is:

$$\mathbb{E}\left[r_t|s_{t-1}, a_{t-1}, s_t\right] = \sum_{r\in\mathcal{R}} r \frac{p\left(s_t, r_t|s_{t-1}, a_{t-1}\right)}{p\left(s_t|s_{t-1}, a_{t-1}\right)} \tag{20}$$

and lastly, the central tenet of the RL agent is its maximisation of cumulative reward over all time t. Therefore, for discrete time RL the cumulative reward function can be defined as:

$$C_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \tag{21}$$

Here, $0 \leq \gamma \leq 1$, $\gamma \in \mathbb{R}$ is a discount factor. This often weights immediate rewards more heavily than future rewards. The discount factor also ensures that the infinite sum converges provided that the sequence of rewards is bounded [8].

As mentioned, the RL agent often has no knowledge about the laws of its environment. The agent can resolve this by learning through experience. It takes actions in the environment which provide a sample of trajectories of rewards, actions and states. Through the collection of these trajectories the agent can learn and improve its policy using the feedback from the reward signal. However, how does the agent compile this information? One approach to a solution method to the RL problem is to frame the expected cumulative reward of the agent as an objective function. The agent can then optimise its policy such that it maximises the expected cumulative reward objective function.[24] If the agent has policy π then, the expected cumulative reward expected under a trajectory, $\tilde{\tau}$, of states, actions and rewards is [8]:

[24] We will see this approach used in the RL solution method called REINFORCE. This is classified as a policy gradient method.

$$J(\theta) = \mathrm{E}_{\tilde{\tau}\sim\pi}\left[\sum_{t} r\left(s_t, a_t\right)\right] \tag{22}$$

As the expectation is evaluated from the sampled experience of the agent, the agent is afforded a means of learning from trial-and-error. Another approach to a solution method for the RL problem is one of approximating the value of the states of the environment. The agent can then select the state which yields the greatest amount of expected reward. In order for the agent to be able to approximate the value of a particular state-action trajectory, we next turn to the Bellman equations.

The Bellman Equations

The following section discusses the *Bellman Equations* [8]. These express the recursive relationship between the value of a state and the expected value of its successor state. The Bellman equations assist RL agents to perform an approximation technique to dynamic programming in the state-action space. The value of a state s_t under an agent's policy π is defined as the expected cumulative reward gained, when starting in state s_t and thereafter following policy π. This is called the *state value function*:

$$V_\pi(s_t) = \mathbb{E}_\pi\left[C_t|t_t\right] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty}\gamma^k r_{t+k+1}|s_t\right], \text{ for all } s \in \mathcal{S}. \tag{23}$$

The value of taking action a_t in state s_t under a policy π is also defined. This is the expected cumulative reward gained, when starting from state s_t, taking action a_t and thereafter following policy π. This is called the *state-action value function*:[25]

$$Q_\pi(s_t, a_t) = \mathbb{E}_\pi\left[C_t|s_t, a_t\right] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty}\gamma^k r_{t+k+1}|s_t, a_t\right] \tag{24}$$

Approximating these value functions forms the basis of many contemporary model-free RL algorithms.[26] The solution methods that focus on approximating the value function are called value-based methods. Since value functions report the expected cumulative reward gained for a state or state-action pair, they therefore directly inform the policy of the agent such that the optimal policy directly corresponds with the optimal value function.

The Bellman equations, defined below, incorporate the value functions in a recursive manner. This adopts a dynamic programming approach which aids the RL solution algorithms' ability to evaluate trajectories in state-action spaces. For a state s_t and policy π, the following condition holds between the value of the state s_t and the value of its successor states, s_{t+1}, for all $s_t \in \mathcal{S}$. This is the *Bellman equation* for the state-value function:

$$\begin{aligned} V_\pi(s_t) &= \mathbb{E}_\pi\left[C_t|s_t\right] \\ &= \mathbb{E}_\pi\left[r_{t+1} + \gamma C_{t+1}|s_t\right] \\ &= \sum_{a_t}\pi(a_t|s_t)\sum_{s_{t+1}}\sum_{r_{t+1}} p\left(s_{t+1}, r_{t+1}|s_t, a_t\right)\left[r + \gamma\mathbb{E}_\pi\left[C_{t+1}|s_{t+1}\right]\right] \\ &= \sum_{a_t}\pi(a_t|s_t)\sum_{s_{t+1}}\sum_{r_{t+1}} p\left(s_{t+1}, r_{t+1}|s_t, a_t\right)\left[r + \gamma V_\pi\left(s_{t+1}\right)\right]. \end{aligned} \tag{25}$$

[25] This is sometimes referred to as the action value function only.

[26] We will discuss the RL taxonomy in terms of model-free and model-based algorithms later, in section 3.2.1.

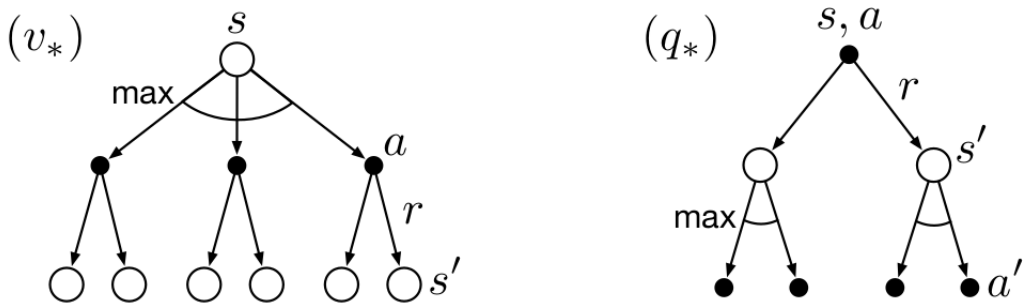

Figure 5: An example of backup diagrams for the optimal value functions V^* and Q^* [8]. This figure displays an agent's possible trajectories across states and actions for an example MDP. The diagram is referred to as a backup diagram where the flow through time runs from the top to bottom. The backup diagram provides useful insight for the way in which the Bellman equations recursively approximate value functions as they show, graphically, the spans of future states and actions that are considered in the Bellman optimality equations for V^* and Q^*. The diagram on the left represents the Bellman optimality equation for V^*, equation 29. The diagram for Q^* represents equation 30.

Figure 5 provides a good intuition for understanding the Bellman equations. First examine the left-hand backup diagram for the optimal state value function, V^*. In this particular example, starting from state s (the root node at the top), the agent could take any of three actions based on its policy. Thereafter the environment could transition to one of several next states, s', while the agent receives an amount of reward, r. These depend on the dynamics given by the probability function, p. The Bellman equation averages over these different possibilities. The Bellman equation specifically states that the value of the starting state of the agent is equal to the expected value of the next state plus the expected reward received along the way. As mentioned, since the agent cannot determine the true state values under a given policy, it instead samples many trajectories under the policy π. It keeps tracks of the rewards received from state-action pairs along these trajectories. This allows it to approximate the state and state-action value functions using *Monte Carlo* expectations from the samples of data. The following defines the *Bellman optimality equations* which define the goal of the RL agent, finding the policy, π, which yields the greatest cumulative reward. Specifically, it is important to note that the value functions define a partial ordering over policies [8] which means that a policy, π, is better than or equivalent to another policy π', if and only if $V_\pi(s) \geq V_{\pi'}(s)$.[27] The optimal policy is denoted here as π^* which directly corresponds to the optimal state value function, V^* and, to the optimal state-action value function Q^*. These are defined, for all $s \in \mathcal{S}$, as [8]:

$$V^*(s) = \max_\pi V_\pi(s) \tag{26}$$

and, for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$ as:

$$Q^*(s,a) = \max_\pi Q_\pi(s,a) \tag{27}$$

where Q^* gives the expected return for taking action a in state s and thereafter following the optimal policy, π^*. Q^* can be written in terms of V^* as [8]:

$$Q^*(s_t, a_t) = \mathbb{E}\left[r_{t+1} + \gamma V^*\left(s_{t+1}\right) | s_t, a_t\right] \tag{28}$$

The diagram on the left of figure 5 represents the Bellman optimality equation for V^* and the diagram on the right represents the Bellman optimality equation for Q^*. One can see from the maximisations in the

[27] This is elaborated on in the next section as the *policy improvement theorem* [8].

diagrams that the optimal value functions result from considering the spans of future states and actions. The Bellman optimality equation for V^* is expressed below. For interpretation this says that the value of a state under an optimal policy, π^*, must equal the expected return for the best action from that state (and thereafter following the optimal policy) [8]:

$$\begin{aligned} V^*(s_t) &= \max_{a_t \in \mathcal{A}(s_t)} Q_{\pi^*}(s_t, a_t) \\ &= \max_{a_t} \mathbb{E}_{\pi^*} \left[C_t | s_t, a_t \right] \\ &= \max_{a_t} \mathbb{E}_{\pi^*} \left[r_{t+1} + \gamma C_{t+1} | s_t, a_t \right] \\ &= \max_{a_t} \mathbb{E} \left[r_{t+1} + \gamma V^* \left(s_{t+1} \right) | s_t, a_t \right] \\ &= \max_{a_t} \sum_{s_{t+1}, r_{t+1}} p \left(s_{t+1}, r_{t+1} | s_t, a_t \right) \left[r_{t+1} + \gamma V^* \left(s_{t+1} \right) \right]. \end{aligned} \tag{29}$$

Similarly, the Bellman optimality equation for Q^* is:

$$\begin{aligned} Q^*(s_t, a_t) &= \mathbb{E} \left[r_{t+1} + \gamma \max_{a_{t+1}} Q^* \left(s_{t+1}, a_{t+1} \right) \middle| s_t, a_t \right] \\ &= \sum_{s_{t+1}, r_{t+1}} p \left(s_{t+1}, r_{t+1} | s_t, a_t \right) \left[r_{t+1} + \gamma \max_{a_{t+1}} Q^* \left(s_{t+1}, a_{t+1} \right) \right] \end{aligned} \tag{30}$$

The Bellman optimality equations provide a system of equations for each state and, for each state-action pair. For finite MDPs where the dynamics of the environment, p, are known then one could (in principle) solve the system of equations for V^* and Q^*. This provides a dynamic programming solution method. In general, for the RL agent, the dynamics of the environment are unknown. As mentioned, the agent resolves this via sampling of trajectories to approximate the value functions via MC expectations. Next we will show that having gained an approximation of the value functions the agent can improve its policy by adopting a *greedy* policy on the updated value functions. Continuously updating the value functions, using the improved greedy policy, will lead to convergence of the approximated value functions to the optimal value functions. Corresponding to this, the process will lead to convergence of the agent's policy to an optimal policy, π^*.

Generalised Policy Iteration

Generalised policy iteration [8] provides a general framework for an algorithm to solve the RL problem. It consists of two processes. These are *policy evaluation* and, *policy improvement*. The generalised policy iteration algorithm can be applied generally to any value function, policy or state-action space, thus providing a general framework for solution methods to the RL problem.

Figure 6 illustrates a generalised policy iteration algorithm. The policy evaluation process approximates the value function of a policy. The algorithm begins with an arbitrary initial value for each state. The Bellman equations are used to iteratively update the value function until convergence. As mentioned before, this uses the MC samples from the agent's experience to estimate expectations. Once the initial policy has been evaluated the agent will perform policy improvement. This process improves the policy by making the agent greedy with respect to the value function of the previous policy.[28] The greedy policy is defined as [8]:

[28] Greedy action selection refers to selecting the action which, according to the value function, will produce the highest reward.

$$\begin{aligned}\pi'(s_t) &= \underset{a_t}{\operatorname{argmax}}\, Q_\pi(s_t, a_t) \\ &= \underset{a_t}{\operatorname{argmax}}\, \mathbb{E}\left[r_{t+1} + \gamma V_\pi\left(s_{t+1}\right) | s_t, a_t\right] \\ &= \underset{a_t}{\operatorname{argmax}} \sum_{s_{t+1} r_{t+1}} p\left(s_{t+1}, r_{t+1} | s_t, a_t\right)\left[r_{t+1} + \gamma V_\pi\left(s_{t+1}\right)\right]\end{aligned} \tag{31}$$

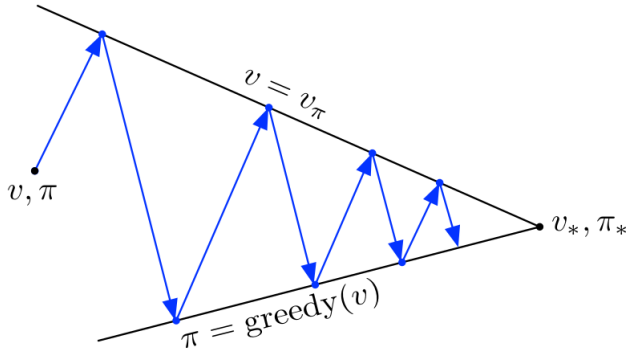

Figure 6: Generalised policy iteration algorithm [8]. This iterates between two processes, policy evaluation and policy improvement. Policy evaluation estimates the value function of states of the environment, under the current policy, π. Policy improvement improves upon the old policy by making the current policy greedy with respect to the updated value function. Iterating between these two processes converges upon an optimal policy and corresponding optimal value function.

As is illustrated by figure 6, an iterative cycle between policy evaluation and policy improvement will eventually converge on the optimal policy π^* and the corresponding optimal value function V^*. The proof of this is based on the policy improvement theorem [8]. This states that if π and π' are any pair of policies such that $Q_\pi\left(s, \pi'(s)\right) \geq V_\pi(s), \forall s \in \mathcal{S}$ then this implies that the policy π' must be better than or equivalent to π.

Two categories of algorithm that employ the iterative procedure of generalised policy iteration to solve the RL problem are Monte Carlo (MC) and Temporal Difference (TD) methods. MC methods were mentioned previously.[29] They require the RL problem to have a terminal state so that complete sample trajectories can be used for state value function estimation. In contrast, TD methods use a bootstrapped estimate from the Bellman equations to resolve this. To estimate the value of a state, TD learning uses the current-time estimate of the value function for its expected successor state (a Bellman recursive update). This is added to the sample average of the reward received along the way. For the case of *Q-learning* (which refers to using the state-action value function as opposed to the state value function, V) this update is [8]:

$$Q\left(s_t, a_t\right) \leftarrow Q\left(s_t, a_t\right) + \alpha\left[r_{t+1} + \gamma \max_a Q\left(s_{t+1}, a\right) - Q\left(s_t, a_t\right)\right] \tag{32}$$

The reward for more than one state can be used before using the bootstrap estimate of the expected successor state. *n-step* TD methods [8], for instance, update the Q values by sampling the rewards for the n steps subsequent to an action and thereafter using the value function estimate for the successor state (which occurs after the n^{th} reward).

The ideas of MC and Q-learning will be referenced in the next section on deep RL methods. Having established a framework for tabular reinforcement learning methods, we can address the problem of scaling by examining two particular methods for *deep reinforcement learning*. These are deep Q networks [9] and REINFORCE, a policy gradient method [10]. These are examined from the context of deep learning.

[29] These use the agent's sampled trajectories to iteratively update the value function estimates for all states based on MC sample averages.

3.2 Deep Reinforcement Learning

The success of RL agents at tackling complex tasks such as Go [35] and the Atari games [9] has been largely due to the approach of deep RL, to make use of neural networks as powerful function approximators. An artificial neural network (ANN) can be viewed as a nonlinear function approximator. It is often used for supervised learning tasks. The *network* aspect of the model describes the graphical view of a neural network, seen below in figure 7. This is illustrated as a directed graph with nodes and edges representing activation functions and parameters of the model, respectively.

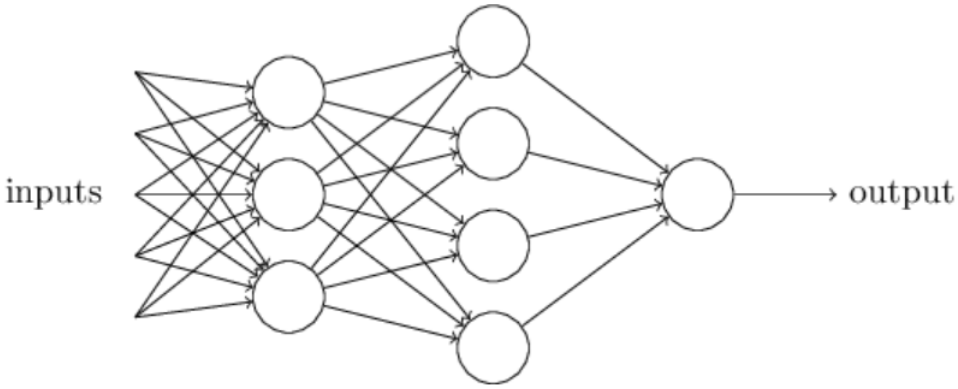

Figure 7: An example of an artificial neural network [37].

Relevant to our study is the *Universal Approximation Theorem* of neural networks [38]. This says that feedforward neural networks are capable of arbitrarily accurate approximation to any real valued continuous function.[30] This theorem provides us with a powerful tool for approximating a vast class of functions. Here, the theorem is applied in order to approximate value functions and, policy functions.

The structure of the network allows the algorithm of backpropagation to be used for evaluating the expression of a gradient of a loss function. This is needed for optimising the network with respect to a set of parameters, θ and labelled data (in the supervised learning context).

The following deep RL algorithms have seen success with the use of deep neural networks as function approximators. Importantly, these are viewed within the model-free paradigm of RL methods. The contrast between this and the model-based paradigm will be discussed later, in section 3.2.1. In this model-free paradigm we discuss policy-based and value-based methods.

REINFORCE: A policy gradient method

Policy-based methods for solving the RL problem focus on parameterising the policy function of an RL agent. The set of parameters, θ, of the policy $\pi(a|s;\theta)$ can be optimised with respect to the objective function seen in equation 22. In other words, the RL agent seeks to find the set of parameters for its policy which accumulate the greatest amount of expected cumulative reward:

$$\theta^* = \underset{\theta}{\text{argmax}}\ J(\theta) \tag{33}$$

By parameterising this function with a deep neural network one can consider this as a problem of gradient ascent. The problem of estimating the expectation can be overcome through the collection of MC samples of trajectories of the agent's experience from a particular policy π. The problem of estimating the gradient

[30] ... that is Borel-measurable.

of the expected cumulative reward function, $J(\theta)$ requires the *Policy Gradient Theorem* [10]. Recall that if the agent has policy π then, the expected cumulative reward expected under a trajectory, $\tilde{\tau}$, of states, actions and rewards is $J(\theta)$. The policy gradient theorem says that the gradient of the objective function can be approximated as:

$$\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta \int d\tilde{\tau} r(\tilde{\tau}) \pi_\theta(\tilde{\tau}) \\
&= \int d\tilde{\tau} r(\tilde{\tau}) \nabla_\theta \pi_\theta(\tilde{\tau}) \\
&= \int d\tilde{\tau} r(\tilde{\tau}) \frac{\nabla_\theta \pi_\theta(\tilde{\tau})}{\pi_\theta(\tilde{\tau})} \pi_\theta(\tilde{\tau}) \\
&= \int d\tilde{\tau} r(\tilde{\tau}) \nabla_\theta \ln \pi_\theta(\tilde{\tau}) \pi_\theta(\tilde{\tau}) \\
&= \mathbb{E}_{\pi_\theta(\tilde{\tau})} \left[r(\tilde{\tau}) \nabla_\theta \ln \pi_\theta(\tilde{\tau}) \right]
\end{aligned} \tag{34}$$

This says that the gradient of the objective function (with respect to the policy parameters) is the average gradient of the policy multiplied by the rewards received. Thus the agent has a means of estimating expectations of reward and performing gradient ascent to improve its policy. This approach is considered scalable due to the use of deep neural networks to approximate the policy function.

Having established the key components that characterise the REINFORCE algorithm, we can classify the algorithm in the RL taxonomy as a *Monte Carlo policy gradient* method. The algorithm below summarises this method. After collecting some MC sample trajectories from a policy, the gradient of the objective function with respect to the policy parameters is computed and stochastic gradient ascent is used for the purpose of updating the policy parameters with some step size hyperparameter α:

Algorithm 1 REINFORCE [10, 39]

Initialisation:

Policy parameters, θ

Policy function $\pi(a|s;\theta)$

Step size parameter α

Reward discount factor γ

for An Episode **do**

Using the current policy, collect trajectories of tuples $(s_t, a_t, r_{t+1}, s_{t+1})$.

for time-step t **do**

Compute some v_t ▷ Either a Monte Carlo reward or some value function

$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{j=1}^{N} \left[\sum_{t=1}^{T} \nabla_\theta \ln \pi(a_t|s_t;\theta) v_t \right]$ ▷ Policy gradient

$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$) ▷ Stochastic gradient ascent

end for

end for

Return $\pi(a|s;\theta)$

Actor-Critic Methods

As MC sampling can be considered a high variance approach due to sample variability, especially for small sample sizes, one can instead make use of the *advantage function* $A_t = r_t - V(s_t)$ in the *actor-critic* architecture [10]. An actor-critic method introduces a value function into the policy gradient approach. This is referred to as the critic as it evaluates the agent's performance and the performance of the current policy. The advantage function, in particular, is one approach at an actor-critic method. The advantage

function above uses the state-value function as a baseline to reduce the variance of the MC estimates. Using the advantage function, the gradient of the policy function appears below, in equation 35, where the advantage function has replaced the reward in equation 34. This gradient can then be used in the MC policy gradient algorithm. This requires the fitting of a value function via gradient descent on a squared-error loss function, where this loss function uses the difference between observed and expected rewards under the current policy $\pi(a|s;\theta)$. The value function fits into algorithm 1 in place of v_t. Next we will review a value function-based method to solving the RL problem using function approximators.[31]

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta(\tilde{\tau})}\left[\underbrace{(r_t - V(s_t))}_{A_t}\nabla_\theta \ln \pi_\theta(\tilde{\tau})\right] \tag{35}$$

Deep Q Networks (DQN): A value-based method

Here we discuss a deep RL solution method that takes the value-based approach [8]. This makes use of the policy improvement theorem[32] which says that if a policy π' has value function with greater values for all states than another policy π's value function then, the policy π' is better than the policy π. Hence, to select a policy we need only approximate value functions, as the agent's actions can be selected such that they are *greedy* with respect to estimated value function. The corresponding policy can be denoted as [1]:

$$\pi(a_t|s_t;\theta) = \begin{cases} 1 & \text{if } a_t = \underset{a\in\mathcal{A}(s_t)}{\text{argmax}}\, Q_\pi(s_t,a_t) \\ 0 & \text{otherwise} \end{cases} = \delta\left(a_t - \max_a Q_\pi(s_t,a_t)\right) \tag{36}$$

Once again deep neural networks are used as function approximators for the value function. For the TD method called Q-learning [8] (equation 32), the state-action value function is approximated. The Q-learning algorithm parameterises the state-action value function with parameters θ. This gives $Q_\theta(s_t,a_t)$. The policy evaluation algorithm is performed by collecting state-action-reward-next state tuples from the agent's experienced trajectories (under some policy). Then the bootstrapped TD value estimates for the Q-values (which appear in equation 32) are collected for the experienced state-action pairs. The mean-squared error loss function can be used to train the parameters of the Q-network. This measures the squared-error loss between the new Q-value estimates and, the estimates from the Q-value function of the previous time-step.[33] Gradient descent updates the parameters, completing the value iteration algorithm. Lastly, a new policy is chosen to be greedy with respect to the updated state-action value function (thus implementing generalised policy iteration).

The above Q-learning algorithm presents three issues [9]: (i) state-action trajectories contain correlated observations which may lead to overfitting in the neural network, (ii) the agent's policy is sensitive to small changes to Q_θ as this changes the loss function which changes the implicit loss surface which the neural network is performing gradient descent on and, (iii) Q_θ is correlated with its target, the previous estimate of Q_θ. To resolve these issues [9] proposed Deep Q Networks (DQN). These involve the use of a random replay buffer which stores a number of state-action-reward-next action tuples from the agent's experienced trajectories and from which the neural network can randomly sample to remove the correlation between observations. Furthermore, they proposed the freezing of a second state-action value function Q_ϕ which remains frozen for a large number of parameter updates of Q_θ. This addresses the issue of correlation between the two value functions and stabilises the loss surface for gradient descent. The resulting DQN algorithm appears below:

[31] Work on asynchronous actors and critics has developed the popular *asynchronous advantage actor-critic* (A3C) method [11].

[32] See the section on generalised policy iteration, 3.1.

[33] This makes Q-learning an off-policy algorithm.

Algorithm 2 Deep Q Networks (DQN) [40]

Initialisation:

Random replay buffer $\mathcal{B}$

State-action value function parameter sets θ, ϕ

State-action value functions Q_θ, Q_ϕ ▷ Principle and target

Fixed number of steps for freezing target network N

for An Episode **do**

Observe s_0

for time-step t **do**

Select action according to current greedy policy $a_t = \text{argmax}_{a_t} Q_\theta(s_t, a_t)$

Execute a_t and observe reward r_t and state s_{t+1}

Store tuple (s_t, a_t, r_t, s_{t+1}) in $\mathcal{B}$

Sample a random minibatch from $\mathcal{B}$

for Each tuple in sampled batch **do**

$$y \leftarrow \begin{cases} r & \text{if } s \text{ is terminal state} \\ r_t + \gamma \max Q_\phi\left(s_{t+1}, a_{t+1}\right) & \text{otherwise} \end{cases}$$

Calculate gradient on MSE loss $\nabla_\theta \mathcal{L} = \nabla_\theta \left(y - Q_\theta\left(s_t, a_t\right)\right)^2$

$\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}$ ▷ Stochastic gradient descent

end for

Every N steps update $\phi \leftarrow \theta$ ▷ Update target network

end for

end for

Return Q_θ

3.2.1 Model-Based RL

Up till now we have reviewed a model-free view of solutions to the RL problem. The model-free approach is used in the case where the agent can sample experiences of states, actions and rewards via its interaction with the environment. The sample averages are then used to estimate the expectations needed for policy-gradient methods and value-based methods. The model-free approach is therefore useful in environments where it is easy and *cost effective* to sample trajectories. By this we mean that the model-free approach is useful in contexts where it does not incur a computational or physical cost such that our agent cannot gain a large sample to estimate expectations from. A context in which a heavy computational cost may occur is an engineering context, for instance with a robotic agent. These contexts form one justification for a model-based approach [1].

The model-based approach refers to the agent forming an explicit model of the state dynamics of its environment [8]. This can be considered a sample efficient approach since the model makes repeated use of limited data. In the model-based RL paradigm, *learning* a model from the agent's experienced trajectories is often accompanied by model-based *planning*.[34] Planning refers to the use of the model to simulate trajectories to improve the agent's policy (or value function). In other words planning takes a model as input, evaluates trajectories using a value function and, outputs an optimised policy.

Model-based RL can learn models using state-action function approximators trained using gradient descent with squared error loss functions and, data from an experience replay buffer, like that used for algorithm 2, DQN. As seen in figure 8 it is often appropriate to use model-based planners to re-plan while training the model of the state dynamics. This aids in the collection of appropriate data such that the model accurately

[34] In this work we focus on the type of model-based algorithm that implements model-based planners. However, other types of model-based algorithm can also be considered. For example model-based value estimation [41] uses fictitious samples from an agent's model to train a model-free policy [42]. The fictitious samples are augmented to real samples for training.

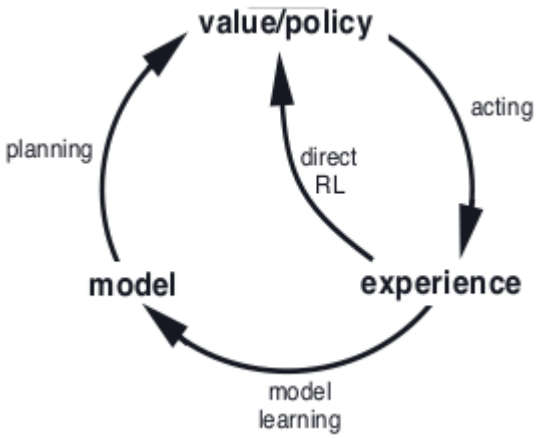

Figure 8: Diagram of a model-based RL algorithm [8].

predicts trajectories that the agent's optimal policy may use. One method of planning is simply selecting the greedy action that maximises expected reward according to the learned model.[35] This approach may ignore the ability of the RL agent to acquire a policy that is best for an entire trajectory of decisions (as opposed to only one decision). Since new experiences that the agent gains whilst learning its model affect the expected reward that the model predicts, it may be best to simultaneously train a model-free style policy at the same time as the model is trained. This approach is also adopted later for the case of active inference agents [17].

Dyna

The Dyna architecture [12] offers a model-based solution approach to the RL problem. The algorithm, appearing below, performs the model-free Q-learning approach seen before. However, the algorithm also learns a model which is used for sampling, thus improving its sample efficiency. The algorithm gives a general framework in which alternative model-free methods and, model-based planners can be used. Note that while not specified below, the model and the Q-network can be parameterised by deep neural networks and trained with gradient descent on some loss function.

Model-Based Planning: The Cross-Entropy Method (CEM)

The cross entropy method (CEM) offers a simple example of a model-based planning method that will be used later when scaling active inference methods in the context of model-based RL methods. As a reminder, model-based planning takes a model as input and outputs an optimised policy. As the model encapsulates the environmental dynamics, policy improvement refers to *finding* the trajectory of actions (the output of the planner) that optimise the agent's cumulative reward.

The CEM [43] is one method of optimising a sequence of actions: $\{a_1, a_2, ..., a_T\}$. The method samples sequences of actions (trajectories) from some initial distribution (which is often multi-variate Gaussian). The optimal sequence is selected based on some value function. Lastly, the distribution of actions (the policy) is updated to select the better trajectories more frequently.

Thus with model-based and model-free algorithms in hand, we have begun to observe how the RL problem can be solved. The deep neural networks have offered success stories in scaling RL agents to larger state-spaces and longer time horizons. Before adopting this approach for active inference agents, it is important to understand how the RL problem can be framed as one of inference. This will guide the reconciliation of active inference with RL and lead to a discussion about exploration and exploitation.

[35] A model of state dynamics may be learnt alongside a reward model, although the reward model is often not needed.

Algorithm 3 Dyna-Q [8]

Initialisation:

Q-Network $Q(s, a)$

Model $f(s, a)$

Random replay buffer $\mathcal{B}$

for Forever **do**

$s \leftarrow$ current state

$a \leftarrow$ greedy action from $Q(s, a)$

Observe reward r and state s'

Store tuple s, a, s', r in $\mathcal{B}$

$Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right]$

Update $f(s, a)$ with r, s'

for n times **do**

$s \leftarrow$ random state from $\mathcal{B}$

$a \leftarrow$ random action from previously taken from s

Sample s', r from $f(s, a)$

Update Q

end for

end for

Return Q

3.3 The RL Problem as Inference

RL can be viewed from the lens of optimal control. The control problem refers to the problem of computing the optimal sequence of actions to fulfil the agent's goals. On the other hand, RL can also be viewed as a problem of inference. In this section we change notation from the tabular RL notation of the previous section to a more general notation in terms of distributions of states, actions and rewards.

Recalling the MDP environment in figure 4, the decision-making agent can take actions, $a \in \mathcal{A}$ which may influence the states, $s \in \mathcal{S}$, of the environment. In the previous section the agent sought to maximise a cumulative reward function. A more general approach is to suppose that the agent has some notion of its preferences and, that these can be expressed as an objective function, a distribution of preferences or as the rewards from the environment [1]. The central tenet of the optimal control problem is to compute the optimal action sequence to achieve the agent's goals.

The agent is assumed to optimise over trajectories of states, actions and rewards, denoted as $\tilde{s}, \tilde{a}, \tilde{r}$. Without loss of generality, we assume that a trajectory is such that $\tilde{s} = \int s(t)dt$ (for the instance of states). The environmental dynamics can be defined in terms of conditional densities of rewards and states: $p(\tilde{s}|\tilde{a})$ and $p(\tilde{r}|\tilde{s})$, similarly to before. This probabilistic formulation allows for the representation of the uncertainty in the environment and the stochasticity of the environment in a unified way. $p(\tilde{r}|\tilde{s})$ represents the reward function. Note that if rewards are deterministic then the reward distribution can be represented by Dirac delta functions. Recall that the Markov property means that both the state and reward depend only on the dynamics at the current time. This gives us $p(\tilde{r}|\tilde{s}, \tilde{a}) = \Pi_t p(r(t)|s(t), a(t))$ for the reward distribution.[36] Finally, the objective function of the RL problem can be expressed as [1]:

[36] In the context of partially observable Markov decision processes (see figure 2) observations are dependent on states: $p(\tilde{o}|\tilde{s})$.

$$\begin{aligned}\underset{a}{\operatorname{argmax}}\ \mathcal{L}_{\text{control}} &= \underset{a}{\operatorname{argmax}} \int p(\tilde{r}|\tilde{s},\tilde{a})p(\tilde{s}|\tilde{a})p(\tilde{a})d\tilde{s} \\ &= \underset{a}{\operatorname{argmax}}\ \mathbb{E}_{p(\tilde{s}|\tilde{a})p(\tilde{a})}[\ln p(\tilde{r}|\tilde{s},\tilde{a})]\end{aligned} \tag{37}$$

Expression 37 can be interpreted as demonstrating that the objective is to select actions that maximise the probability or amount of reward expected under the trajectory of states given the agent's policy of actions.[37]

Model-Based RL: A Probabilistic Formulation

One can denote the agent's model of state transition dynamics by the variational probability density $q_\phi(\tilde{s}|\tilde{a})$. If the agent is in a context where the reward function is unknown, then it can model the reward function which is denoted by the probability density $q_\theta(\tilde{r}|\tilde{s},\tilde{a})$.[38] These represent models of the true environmental transition dynamics and, the true reward function which we denote as $p(\tilde{s}|\tilde{a})$ and $p(\tilde{r}|\tilde{s},\tilde{a})$, respectively. Referring to the objective function in equation 37 we use importance sampling to introduce these models [1]:

$$\begin{aligned}\underset{a}{\operatorname{argmax}}\ \mathcal{L}_{\text{control}} &= \underset{a}{\operatorname{argmax}}\ \mathbb{E}_{p(\tilde{s}|\tilde{a})p(\tilde{a})}[\ln p(\tilde{r}|\tilde{s},\tilde{a})] \\ &= \underset{a,\phi,\theta}{\operatorname{argmax}}\ \mathbb{E}_{p(\tilde{s}|\tilde{a})p(\tilde{a})}\frac{q(\tilde{s}|\tilde{a})}{q(\tilde{s}|\tilde{a})}\left[\ln p(\tilde{r}|\tilde{s},\tilde{a})\frac{q(\tilde{r}|\tilde{s},\tilde{a})}{q(\tilde{r}|\tilde{s},\tilde{a})}\right] \\ &= \underset{a,\phi,\theta}{\operatorname{argmax}}\ \mathbb{E}_{q(\tilde{s}|\tilde{a})p(\tilde{a})}\frac{p(\tilde{s}|\tilde{a})}{q(\tilde{s}|\tilde{a})}\left[\ln q(\tilde{r}|\tilde{s},\tilde{a})\frac{p(\tilde{r}|\tilde{s},\tilde{a})}{q(\tilde{r}|\tilde{s},\tilde{a})}\right] \\ &= \underset{a,\phi,\theta}{\operatorname{argmax}}\ \mathbb{E}_{q(\tilde{s}|\tilde{a})p(\tilde{a})}\ln\frac{p(\tilde{s}|\tilde{a})}{q(\tilde{s}|\tilde{a})}\left[\ln q(\tilde{r}|\tilde{s},\tilde{a})\frac{p(\tilde{r}|\tilde{s},\tilde{a})}{q(\tilde{r}|\tilde{s},\tilde{a})}\right] \\ &= \underset{a,\phi,\theta}{\operatorname{argmax}}\ \underbrace{\mathbb{E}_{q(\tilde{s}|\tilde{a})p(\tilde{a})}[\ln q(\tilde{r}|\tilde{s},\tilde{a})]}_{\text{Reward Maximisation}} - \underbrace{D_{KL}[q(\tilde{s}|\tilde{a})\|p(\tilde{s}|\tilde{a})]}_{\text{System Identification}} - \underbrace{\mathbb{E}_{q(\tilde{s}|\tilde{a})p(\tilde{a})}\left[\ln\frac{q(\tilde{r}|\tilde{s},\tilde{a})}{p(\tilde{r}|\tilde{s},\tilde{a})}\right]}_{\text{Reward Model Identification}}\end{aligned} \tag{38}$$

The three terms appearing above offer insight as to the maximisation of the objective function. The first term highlights the maximisation of expected reward under the environmental dynamics which the agent's model expresses. The second objective, called system identification, is a minimisation objective whereby the agent wants to minimise the difference between its model of the environment and the true environmental probability transition dynamics. Similarly, the third term expresses the objective of minimising the difference between the agent's model of the reward dynamics and the true reward dynamics. The second term is minimised with respect to the parameters of the agent's model of environment dynamics and, the third term is minimised with respect to the parameters of the agent's model of the reward function. As the minimisation occurs over all three terms, the parameters of the agent's models for the environmental dynamics and reward function are optimised over the reward maximisation term. This leads to a bias of the parameters of these models as they are biased toward dynamics which lead to greater amounts of reward. Similarly, the selection of optimal policies is biased toward selecting actions which minimise the divergence between the true and modelled environmental dynamics. This poses the issue of the model bias of model-based RL methods [20]. To resolve this the three terms in equation 38 can be optimised independently [1]:

[37] Note that use of the natural logarithm is permissible as it is a monotonic increasing function and improves numerical optimisation procedures [1].

[38] This is not the case in many RL contexts, where the reward is well specified.

$$
\begin{aligned}
&\underset{a}{\text{argmax}}\ \mathbb{E}_{q(\tilde{s}|\tilde{a})p(\tilde{a})}[\ln q(\tilde{r}|\tilde{s},\tilde{a}) \\
&\underset{\phi}{\text{argmin}}\ D_{KL}\left[q_\phi(\tilde{s}|\tilde{a})\|p(\tilde{s}|\tilde{a})\right] \\
&\underset{\theta}{\text{argmin}}\ \mathbb{E}_{q(\tilde{s}|\tilde{a})p(\tilde{a})}\ln\frac{q(\tilde{r}|\tilde{s},\tilde{a};\theta)}{p(\tilde{r}|\tilde{s},\tilde{a})}
\end{aligned}
\tag{39}
$$

3.3.1 Policy Search as Inference

The RL objective function in equation 37 provides a probabilistic formulation for the objective function. In this section this is turned into an inferential objective function, thereby relating the RL problem to Bayesian inference. Following [20], one can see that this generalisation of the RL problem, which corresponds to maximum entropy RL, also corresponds to exact probabilistic inference for the case of deterministic dynamics and, variational inference[39] in the case of stochastic dynamics.

In order to frame the control problem as one of inference [20] defines binary random variables $\Omega_{1:T}$. These represent whether a given trajectory is optimal or not, with 1 representing optimality. As a result of defining these binary variables, inferring the optimal policy can now be referred to as finding the density $p(\tilde{a},\tilde{s}|\tilde{\Omega}=1)$. Interestingly, this "optimal control as inference" can be represented by a probabilistic graphical model (figure 9).

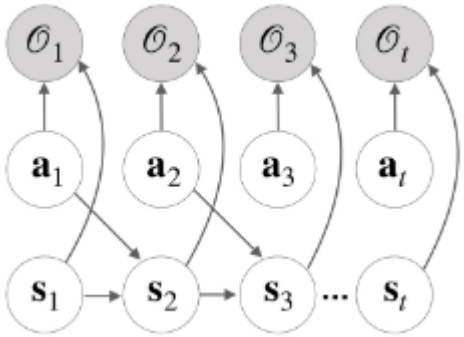

Figure 9: A probabilistic graphical model of an MDP with added hidden optimality variable nodes, $\mathcal{O}$ (or Ω in main text). Conditioning on the optimality variables being true and then inferring the most probable action sequence, corresponds to approaching the RL problem as a problem of inference. The figure appears in [20].

Additionally to the optimality nodes, the following assumption of proportionality is imposed:

$$
p\left(\Omega_t = 1|a_t, s_t\right) \propto \exp\left(r\left(s_t, a_t\right)\right) \tag{40}
$$

This assumption gives us the result that the reward maximisation is equivalent to maximum likelihood estimation due to the log-likelihood of optimality being equal to the reward function. Next, Bayes law is used to provide an expression for the policy [20]. Here $\tilde{\Omega}$ refers to the random variable representing the optimality of all future states in a trajectory.

[39] See section 2.2.

$$
\begin{aligned}
p\left(a_t|s_t,\tilde{\Omega}\right) &= \frac{p\left(\tilde{\Omega},s_t,a_t\right)}{p\left(\tilde{\Omega},s_t\right)} \\
&= \frac{p\left(\tilde{\Omega}|s_t,a_t\right)p\left(a_t|s_t\right)p\left(s_t\right)}{p\left(\tilde{\Omega}|s_t\right)p\left(s_t\right)} \\
&= \frac{p\left(\tilde{\Omega}|s_t,a_t\right)p\left(a_t|s_t\right)}{p\left(\tilde{\Omega}|s_t\right)} \\
&\approx \frac{p\left(\tilde{\Omega}|s_t,a_t\right)}{p\left(\tilde{\Omega}|s_t\right)}
\end{aligned}
\tag{41}
$$

Note that a uniform distribution is assumed for the prior of actions given the states. This gives $p(a_t|s_t) \propto 1$.[40] The numerator of the expression on the last line, $p\left(\tilde{\Omega}|s_t,a_t\right)$, can be interpreted as the probability of optimality of all future states, given the current state and action. This is therefore related to the state-action value function (Q-value, see algorithm 2). Similarly to the numerator, the denominator corresponds to the state-value function, V. Thus, due to the assumption of proportionality between the reward function and distribution of optimality (expression 40) we have a direct relation between the value functions and the natural logarithm of the optimality probability [20]:

$$
\begin{aligned}
Q &= \ln p\left(\Omega_{t:T}|s_t,a_t\right) \\
V &= \ln p\left(\Omega_{t:T}|s_t\right)
\end{aligned}
\tag{42}
$$

The use of Bayes law, above, combined with this relationship to the value functions means that this provides an alternative representation of the recursive Bellman equations [20]: [41]

$$
p\left(\Omega_{t:T}|s_t,a_t\right) = \int ds_{t+1}da_{t+1}p\left(\Omega_{t+1:T}|s_{t+1},a_{t+1}\right)p\left(s_{t+1},a_{t+1}|s_t,a_t,\Omega_t\right)p\left(\Omega_t|s_t,a_t\right) \tag{43}
$$

The value functions, in expression 42, can be referred to as "soft" maxima which correspond to the hard maxima in the original Bellman equations (expression 30). This is since the log-space signal (from Q or V) correspond to "soft" variants of the value functions. Again using the uniform distribution assumption, $p(a_t|s_t) \propto 1$ for the prior of actions given states, the state value function can be seen as the natural logarithm of the marginalisation over actions of the action value function. Furthermore, given large Q values this corresponds to a hard maximum over a_t [20]:

$$
V\left(s_t\right) = \ln \int_{\mathcal{A}} \exp\left(Q\left(s_t,a_t\right)\right)da_t \approx \max_{a_t} Q\left(s_t,a_t\right) \tag{44}
$$

For small Q values the maximum is soft hence the V and Q functions can be referred to as soft value functions. Note that taking the natural logarithm of equation 43 yields the Bellman equation for Q, in the case of deterministic dynamics [20]:

$$
Q\left(s_t,a_t\right) = r\left(s_t,a_t\right) + V\left(s_{t+1}\right) \tag{45}
$$

[40] This assumption is without loss of generality as the non-uniform $p(a|s)$ can be incorporated into the $p(\Omega|s,a)$ via the reward function [20].

[41] Previously, the Bellman equations appeared in section 3.1.

The case of stochastic dynamics gives:

$$Q\left(s_t, a_t\right) = r\left(s_t, a_t\right) + \ln \mathbb{E}_{p\left(s_{t+1}|s_t,a_t\right)}\left[\exp\left(V\left(s_{t+1}\right)\right)\right] \tag{46}$$

As the stochastic dynamic equation for Q takes the soft maximum of the next expected state value, it has the effect of embedding an optimism bias in the Q value. This means that if one possible outcome for the next state, even if unlikely, has a particularly high possible value it will dominate the backup [20]. The implication for the behaviour of the agent is that it is risk-seeking, taking actions that may have extreme risk. This means that this formulation is not suited to stochastic dynamics. However the new, "soft" definitions of value functions satisfy the Bellman recursive relationships and can thus be used to derive the RL algorithms seen previously, such as Q-learning.

Variational Inference Approach

A second approach to formulating the RL problem as inference is to use variational inference [20]. Using the insights of variational inference from section 2.2, we can try to approximate the true posterior distribution for states and actions given optimality, $p(\tilde{a}, \tilde{s}|\tilde{\Omega})$, using a variational distribution $q(\tilde{s}, \tilde{a})$. Following the variational inference approach means attempting to minimise the KL divergence between these distributions.

$$\begin{aligned} & D_{\mathrm{KL}}[q(\tilde{s}, \tilde{a}) \| p(\tilde{s}, \tilde{a}|\tilde{\Omega})] \\ =& D_{\mathrm{KL}}\left[q(\tilde{s}, \tilde{a}) \| \frac{p(\tilde{s}, \tilde{a}, \tilde{\Omega})}{p(\tilde{\Omega})}\right] \\ =& \underbrace{D_{\mathrm{KL}}[q(\tilde{s}, \tilde{a}) \| p(\tilde{s}, \tilde{a}, \tilde{\Omega})]}_{\text{Free Energy}} + \ln p(\tilde{\Omega}) \end{aligned} \tag{47}$$

Since the $\ln p(\tilde{\Omega})$ is constant with respect to the variational density $q(\tilde{s}, \tilde{a})$, we need only minimise the variational free energy (VFE) term. The VFE term can be expressed as the objective function [1]:

$$\begin{aligned} \mathcal{L} &= D_{\mathrm{KL}}[q(\tilde{s}, \tilde{a}) \| p(\tilde{s}, \tilde{a}, \tilde{\Omega})] \\ &= D_{\mathrm{KL}}[q(\tilde{a}|\tilde{s}) q(\tilde{s}) \| p(\tilde{\Omega}|\tilde{s}, \tilde{a}) p(\tilde{a}|\tilde{s}) p(\tilde{s})] \\ &= \underbrace{\mathbb{E}_{q(\tilde{s},\tilde{a})}[\ln p(\tilde{\Omega}|\tilde{s}, \tilde{a})]}_{\text{Reward Maximisation}} + \underbrace{D_{\mathrm{KL}}[q(\tilde{a}|\tilde{s}) \| p(\tilde{a}|\tilde{s})]}_{\text{Action Divergence}} + \underbrace{D_{\mathrm{KL}}[q(\tilde{s}) \| p(\tilde{s})]}_{\text{Dynamics Divergence}} \end{aligned} \tag{48}$$

The variational inference formulation for the objective function of course bears similarity to the general RL objective (expression 37) apart from the extra two terms. Two assumptions are applied to modify this objective function [1]. First, the variational dynamics are assumed to be equal to the true state dynamics $q(\tilde{s}) = p(\tilde{s})$ (as the agent cannot change the dynamics except through action). The state dynamics divergence term, therefore, vanishes. Second, the assumption of proportionality between the distribution of the trajectory of binary optimality variables and the reward function, $p(\tilde{\Omega}|\tilde{s}, \tilde{a}) = \prod_t \exp\left(r\left(a_t, s_t\right)\right)$, is repeated. This gives [1, 20]:

$$\mathcal{L} = \underbrace{\mathbb{E}_{q(\tilde{a}|\tilde{s})p(\tilde{s})}\left[\sum_t^T r\left(s_t, a_t\right)\right]}_{\text{Reward Maximisation}} + \underbrace{D_{\mathrm{KL}}[q(\tilde{a}|\tilde{s}) \| p(\tilde{a}|\tilde{s})]}_{\text{Action Divergence}} \tag{49}$$

The appearance of this objective function is thus substantially closer to the traditional RL objectives, except for the action divergence term between the variational action distribution (which is the agent's policy) and the prior action distribution. Given some prior, this term draws the agent's variational policy close to its policy prior. However, assuming that this prior is uniformly distributed yields [1, 20]:

$$\begin{aligned}\mathcal{L} &= \mathbb{E}_{q(\tilde{a}|\tilde{s})p(\tilde{s})}\left[\sum_{t}^{T} r\left(s_t, a_t\right) - \ln q\left(a_t|s_t\right)\right] \\ &= \mathbb{E}_{q(\tilde{a}|\tilde{s})p(\tilde{s})}\left[\sum_{t}^{T} r\left(s_t, a_t\right)\right] + \mathcal{H}[q(\tilde{a}|\tilde{s})]\end{aligned} \tag{50}$$

This gives an objective function which maximises reward and, at the same time, maximises the entropy of the variational policy $q(\tilde{a}|\tilde{s})$. This is called *maximum entropy RL* [20]. By taking the gradient of this objective with respect to its parameters one can optimise the policy in what can be considered a traditional model-free policy-gradient approach. The difference between this and the traditional policy-gradient methods is that the maximisation of entropy also incorporates an implicit balance between the exploration and exploitation of the agent. The presence of a regularisation term keeps the learned policy $q(\tilde{a}|\tilde{s})$ as close as possible to a prior over actions $p(\tilde{a}|\tilde{s}$. As we have seen, setting this prior to a (diffuse) uniform prior over actions results in the regularisation term becoming a maximum entropy term for the policy. This ensures some random exploration occurs while still maintaining the reward maximisation objective. However, as will be discussed below, random exploration may be sub-optimal in a sparse reward environment. In this instance more directed information-seeking objectives may be useful. Active inference agents, through their objective functions provides this benefit. We will later see maximum entropy RL used in [17].

3.4 The Exploration and Exploitation Dilemma

The exploration exploitation dilemma [7, 8] refers to the trade-off between the agent exploring new regions of its state-space and exploiting its knowledge about which trajectories of the state-space yield the highest rewards. The problem arises in contexts of large state-spaces and, long time horizons. Here, learning the optimal policy can prove challenging because of the difficulty of searching the whole space. Furthermore, exploring the extent of the space is necessary for improved model accuracy. The challenge is that if the agent does not exploit its knowledge, it incurs an opportunity cost associated with the gain of (potentially) optimal reward [1]. Conversely, if the agent exploits its knowledge and adopts the trajectory which according to its model or policy achieves the optimal reward then it may incur a constant opportunity cost, proportional to the distance to the global optimum. This incurs a cost for the agent every time it exploits its knowledge of a local optimum. However, if the agent possesses knowledge of a local optimum that is sufficiently optimal then in large state-spaces almost all exploration will be worse-off. Thus exploration is often kept to a minimum to maximise return in the long run.

One simple approach to addressing this trade-off is called the ϵ-greedy approach [8]. This assigns a constant probability, ϵ, of choosing a random action in order for the agent to explore its environment. Therefore, with a probability of $1 - \epsilon$, the agent will exploit its current knowledge and take the *greedy* action, thereby increasing its knowledge of that particular trajectory. The value of ϵ is typically small (with a range of 0.02-0.05 suggested in [1]). Many implementations of the ϵ-greedy approach decay ϵ with time in order to encourage convergence to an optimum. This means that the agent explores decreasing amounts of the state-space as it improves its policy or model.

A second approach is called Boltzmann exploration [44] . This is a probabilistic approach to exploitation whereby actions are selected from a softmax distribution of their state-action value function (Q-values). A parameter, β, is used to control the entropy of the distribution, whereby a large β value yields a lower entropy and less exploration and, a smaller β will imply a greater exploratory drive. This stochastic policy

appears as:[42]

$$q(a|s) = \frac{\beta \exp(-Q(a|s))}{\sum_a \exp(-Q(a|s))} \tag{51}$$

Similarly to Boltzmann exploration, adding a maximum entropy term to the objective function achieves an exploratory means, as we saw for the variational inference formulation of the RL problem above [20]. Instead of adding a hyper-parameter to the policy, this approach adds an entropy term to the objective function (similarly to the active inference objective function). Therefore, instead of the agent solely maximising reward, it maximises reward while keeping the entropy of its policy at a maximal level. While this can still be categorised as a random exploratory approach, it differs from the ϵ-greedy approach because of the explicit trade-off between random exploration and reward maximisation and because of its explicit derivation from the variational inference approach to control as inference [20].

The above approaches are categorised as means of random exploration where actions are selected at random to drive exploration of the state-space. This presents the problem that these methods are necessarily inefficient, performing particularly poorly in sparse-reward environments where the rewards are hard to obtain and may require a particular policy to achieve a reward at all [14]. Other methods that employ more *directed* objectives for information gain can be employed [1] to ameliorate the inefficiencies of random exploration. For instance, the *direction* of information gain of an active inference agent means that the agent seeks to explore only those areas of the state-space in which uncertainty can be resolved. The investigation about how such an exploratory drive benefits the agent and how this compares to other exploratory methods is not studied in detail hereafter, however this remains an area for future investigation.

With the exception of the maximum entropy approach to exploration which was derived from the variational inference approach to the RL problem, the other approaches can be considered ad-hoc in application, meaning that their mathematically principled origin remains questionable [1]. This offers another benefit of the active inference exploratory objective as it is explicitly derived from the minimisation of variational free energy. Thus the framework gives a principled justification for its application.

An RL Agent Example

To conclude this section we describe an example of a commonly used RL environment, from Open AI Gym [23]. This example is used to benchmark one of the three scaled active inference algorithms (in [14]), which we review in the next section.[43] The environment, depicted in figure 11, requires the agent to drive a car up a one dimensional hill to reach the yellow flag at the top. However, the car's engine is not strong enough to reach the top on its own. The agent must, therefore, learn to drive up the hill on the left to gather enough momentum to drive to its goal. In the implementation of [14], the agent holds a one dimensional, continuous action space corresponding to the strength and direction that it drives. The environment is expressed in terms of a two-dimensional state-space, corresponding to the agent's position and velocity. Moreover, in the implementation of [14], the agent is presented with a reward of $+1$ when it reaches its goals and 0 otherwise. This presents a challenging, sparse-reward reward environment for the RL agent. However, due its directed, intrinsic exploratory drive this provides a context in which the active inference agent thrives [14].

With this particular motivation for the study of active inference agents in the context of the RL problem, with the tools of deep learning and, with the framing of the RL problem as one of inference, we next turn to studying the application of the insights of deep RL to scale active inference agents.

[42] Interestingly this equation corresponds with the softmax of the EFE function. Hence if one considered the EFE as a value function, one can consider the active inference agent as performing Boltzmann exploration, controlled by its β hyperparameter. However the exploration embedded in the EFE objective function embeds a directed exploration motivation in the agent.

[43] ... although the proof-of-concept trial is not discussed in this report.

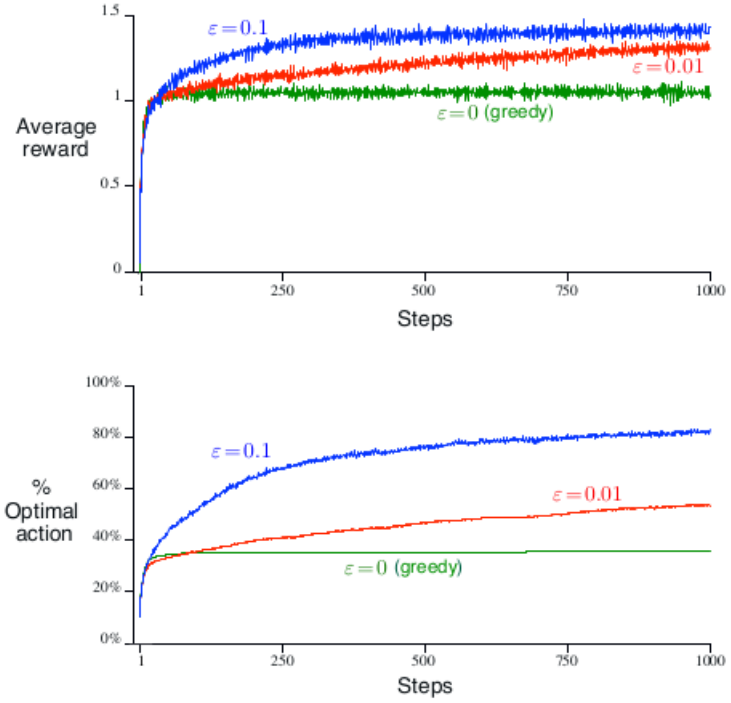

Figure 10: An example of the performance of several ϵ-greedy agents at the multi-armed bandit problem from [8]. The problem involves a choice between several unknown static reward-generating distributions. At each time-step the agent can select one of these reward-generating distributions from which it receives a random reward. It is then able to update its internal value function, which assigns a value to all the distributions (from the agent's perspective). This function assigns values according to the agent's beliefs (about which distribution generates greater amounts of reward). If the agent starts (at some time point) from the perspective that one of the distributions is better than the others, then exploitation refers to selecting the action that picks *that* particular distribution to receive a reward from. Random exploration from $\epsilon \neq 0$ offers the agent a performance benefit since it is able to more readily explore distributions from which it has not yet sampled and, therefore, update its beliefs (top). However, constant proportions of random exploration mean that some proportion of actions will always be non-optimal thus demonstrating the inefficiency of random exploration (bottom).

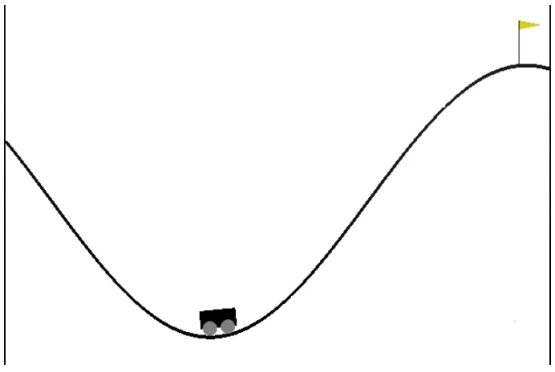

Figure 11: An illustration of the mountain car environment from OpenAI Gym [23].

4 Scaling Active Inference with Deep RL Methods

Thus far we have seen that active inference and RL agents both address the problem of selecting an optimal action sequence that maximises some reward function (or notion of preferences) in an uncertain environment. Active inference agents offer the potential benefit of a directed intrinsic exploratory motivation which is embedded in their objective function, the minimisation of free energy. Furthermore, the benefits of using deep RL algorithms has seen widespread success at scaling tabular RL methods to achieve strong performance at complex tasks. Since both RL and active inference agents can be seen to address the same problem, it would be useful to take the insights of both and combine them. In this section, the problem of scaling active inference agents is addressed via the discussion of three contemporary active inference algorithms [13, 14, 17].[44]

Active inference agents are limited by their scalability. Many past approaches have limited active inference agents to known, small discrete state-action spaces [1, 15, 6, 16]. These approaches operate using categorical distributions for the discrete variables. Additionally, they often assume knowledge of the likelihood, $p(o|s)$, and prior, $p(s)$, matrices.[45] In these tabular approaches the agent's policy is often taken to be the softmax distribution of the expected free energy (EFE) values. The evaluation of the EFE path integral is required for all trajectories in order to derive the agent's policy. This evaluation scales exponentially in terms of both the size of the state-space and the length of the time horizon [1]. As discussed, neural networks can be used to effectively approximate the agent's policy and value functions. Algorithms for the model-free RL agent have taken this approach. These are useful in situations with large state-action spaces [8]. Hence here, we review attempts at using neural networks for the purpose of overcoming problems of scalability for the active inference agent [1, 13, 14, 17].

In the previous section we discussed a formulation of RL using variational inference principles. This posed the RL problem as one of inference. In this section we study a construction of the active inference agent as an RL agent, with the objective to minimise variational free energy (VFE). The neural networks in the case of the active inference agent can be used to approximate the key probability density functions which appear in the expression for VFE. A neural network is also used for the evaluation of the EFE, for the purpose of action selection. Two main approaches are discussed. These are inspired by the model-free [13] and model-based [14] RL paradigms. A third approach hybridises these [17].

As mentioned, the model-free approach [13] uses neural networks to approximate the key distributions of the VFE objective function. The parameters of the networks are trained to jointly minimise VFE. While applying the recursive principles of the Bellman equation along with Q-learning, a bootstrapped value network is used to approximate the EFE value function. This furnishes the algorithm with an actor-critic style of architecture. The model-based approach [14], uses model-based-iterative planning to improve the variational action distribution (the policy). This defines a new objective function, called the *Free Energy of Expected Future*. When minimised, this yields that the variational policy is the softmax of a free energy path integral through time.

Lastly, a hybridised approach [17] defines the paradigm of model-free and model-based learning under a new taxonomy for RL algorithms [21]. This taxonomy consists of two orthogonal dimensions for categorising RL algorithms. The first dimension of categorisation asks whether a particular algorithm infers an optimal sequence of actions (for instance with the CEM planner) or, whether it infers one action at a time (for example in the case of greedy policies). The second dimension of categorisation labels two types of inference, iterative and amortised inference. In terms of the variational policy posterior, these correspond to asking whether the variational parameters are optimised directly via an iterative updating procedure upon the arrival new data (for instance with a CEM planner) or, whether the parameters of the policy posterior are updated via an amortisation function (which generalises across a dataset to amortise the cost of inference).

[44] The benefits of the directed exploratory drive of the active inference agent's objective function is left as an open question and is an ongoing topic of research [13, 14].

[45] In optimal control settings these cannot simply be assumed.

Upon elaboration of the benefits of these types of inference, a hybridised inference approach is discussed [17]. This uses a model-free policy (an amortised inference approach) to provide the starting point for a model-based iterative planning procedure. This procedure draws from the benefits of both model-free and model-based methods, acquiring both sample efficiency and asymptotic performance. Thus, given these three approaches (model-free, model-based and hybridised) we can address the problem of scaling active inference agents using deep RL inspired methods.

4.1 Variational Policy Gradients: A Model-Free RL Approach to Active Inference

Recall that the *tabular active inference agent* [13] possesses a generative model of its environment. Here, this includes states, observations and actions:

$$p\left(o_{1:T}, s_{1:T}, a_{1:T}\right) = p\left(o_1, s_1\right) p\left(a_1\right) \prod_{t=2}^{T} p\left(o_t|s_t\right) p\left(s_t|s_{t-1}, a_{t-1}\right) p\left(a_t|s_t\right) \tag{52}$$

The agent's goal is to model a posterior distribution for states and actions given observations, $p(s, a|o)$, using its generative model.[46] It does this via its imperative to minimise VFE (the KL divergence between its variational distribution, $q(s, a|o)$, and its generative model). As has been discussed, in the tabular active inference approach, the agent's policy is estimated from the softmax distribution of the enumerated EFE of each possible policy trajectory. The total EFE for a policy corresponds to the sum of EFE for each time-step up to the time horizon (the ergodicity assumption), with the expectation being taken using the (current-time) variational posterior. This enumeration of all possible trajectories requires a collection of the agent's experienced states, actions and rewards via the agent's generative model and, from the current time until the time horizon. As the state-action space and the time horizon grows, this enumeration scales exponentially, presenting a limitation of scale [1].

However (and as mentioned), the analogous tabular RL problem has been successfully scaled with the use of deep neural networks [8]. The proposed approach of scaling active inference agents, in this section, is inspired by the model-free paradigm of the RL problem [13]. Specifically, this approach approximates the probability densities of the above generative model with deep neural networks. This allows the agent to learn its environmental dynamics as opposed to using pre-specified distributions. The parameters of the networks are trained together using gradient descent on the agent's VFE objective function (as the agent seeks to minimise this quantity to perform approximate Bayesian inference).[47]

When defining the active inference agent here, our approach looks to recast the RL problem as one of inference (as to the optimal state-action distribution).[48] This is formalised using a discrete time POMDP (see figure 2). The agent's POMDP is described by the generative model above, equation 52 [13]. The agent is assumed to know the general structure of the generative model, although not any details as to the model's distributions [1]. The agent is also assumed to hold a prior preference distribution such that it prefers to receive greater amounts of reward. This can be formulated (for the case of observations) as [13]:

$$\tilde{p}\left(o_{1:T}\right) = \prod_{t=1}^{T} \sigma\left(r\left(o_t\right)\right) \tag{53}$$

where σ refers to a softmax function seen in expression 11. This repeats the assumption used in the 'RL as inference' section, section 3.3.

[46] ... which in the tabular approach is often fully specified, as seen in the example of section 2.4.
[47] See the section on variational inference, section 2.2.
[48] ... thus taking a similar approach to the section on control as inference, section 3.3.

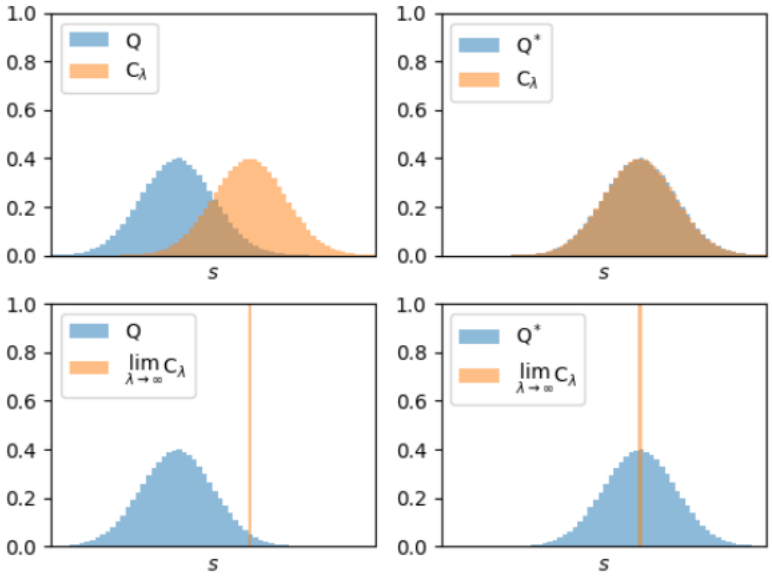

Figure 12: This example is posed by [6]. If one approaches an MDP environment (in which the states are known directly) one can include a parameter λ (often called the temperature parameter). λ controls the spread of the prior preference distribution, where this is $\tilde{p} = C_\lambda = \prod_{t=1}^{T} \sigma(\lambda r(s_t))$. Additionally if it is assumed that the variational distribution $q(s_t|a_{t-1}, s_{t-1}) = Q$, then the figure depicts how an active inference agent selects actions such that the variational distribution $Q(s_t|a_{t-1}, s_{t-1})$ most closely matches the preference distribution C_λ. In this example the preferences and variational distribution are assumed to have a Gaussian shape, with the variational distribution having a fixed variance with respect to action sequences such that the only parameter that is optimised by action selection is the mean of the Gaussian. As described by [6], in the limit, $\lim_{\lambda\to\infty}$, C_λ becomes a Dirac delta distribution. This corresponds to the only optimal trajectory (bottom left and bottom right). Thus minimising the EFE corresponds to selecting the action such that the variational distribution for states will assign the most mass to the reward-maximising state (bottom right). Q^* denotes the optimal predictive distribution over states given the action sequence that minimises EFE (the action sequence $\tilde{a}^* = \text{argmin } \mathcal{G}$). Hence, this figure illustrates the *self-evidencing* characteristic of active inference agents and, more generally, the free energy principle.

The agent seeks to infer the posterior state-action distribution, given the observations of the environment, $p(s_{1:T}, a_{1:T}|o_{1:T})$. This encapsulates its policy. Following the arguments from variational inference, the computation of this state-action posterior distribution is intractable. Hence, an approximate posterior (variational distribution) is used instead, $q(s_{1:T}, a_{1:T}|o_{1:T})$. In this case, the variational inference procedure is followed by minimising the KL divergence between the variational and true posterior. Since this divergence measure is intractable (due to the presence of the true posterior in the expression) the agent instead minimises the VFE which provides a tractable bound on the desired KL divergence measure. The VFE, used here, is defined below as [1, 13]:

$$\begin{aligned}\mathcal{F}(o_{1:T}) &= D_{\mathrm{KL}}\left[q(a_{1:T}, s_{1:T}|o_{1:T}) \,\|p(s_{1:T}, a_{1:T}, o_{1:T})\right] \\ &= KL\left[q(a_{1:T}, s_{1:T}|o_{1:T}) \,\|p(s_{1:T}, a_{1:T}|o_{1:T})\right] + \ln p(o_{1:T}) \\ &\geq KL\left[q(a_{1:T}, s_{1:T}|o_{1:T}) \,\|p(s_{1:T}, a_{1:T}|o_{1:T})\right]\end{aligned} \tag{54}$$

Decomposing this posterior for interpretation and following [1], we can recover the accuracy and complexity terms seen in equation 12 along with an additional term:

$$
\begin{aligned}
\mathcal{F}\left(o_{1:T}\right) &= D_{\mathrm{KL}}\left[q\left(a_{1:T}, s_{1:T}|o_{1:T}\right) \| p\left(s_{1:T}, a_{1:T}, o_{1:T}\right)\right] \\
&= \underbrace{-\mathbb{E}_{q(a_{1:T}, s_{1:T}|o_{1:T})}\left[\ln p\left(o_{1:T}|s_{1:T}\right)\right]}_{\text{Accuracy}} + \underbrace{\mathbb{E}_{q(a_{1:T}|s_{1:T})} D_{\mathrm{KL}}\left[q\left(s_{1:T}|o_{1:T}\right) \| p\left(s_{1:T}\right)\right]}_{\text{Complexity}} \\
&+ \underbrace{D_{\mathrm{KL}}\left[q\left(a_{1:T}|s_{1:T}\right) \| p\left(a_{1:T}|s_{1:T}\right)\right]}_{\text{Action Divergence}}
\end{aligned} \tag{55}
$$

Further, applying the Markov property to the posterior, $q\left(a_{1:T}, s_{1:T}|o_{1:T}\right) = \prod_{t=1}^{T} q\left(a_t|s_t\right) q\left(s_t|o_t\right)$ gives:

$$
\begin{aligned}
\mathcal{F}\left(o_{1:T}\right) &= \sum_{t=0}^{T} \mathcal{F}_t\left(o_t\right) = \sum_{t=0}^{T} D_{\mathrm{KL}}\left[q\left(a_t, s_t|o_t\right) \| p\left(s_t, a_t, o_t\right)\right] \\
&= \sum_{t=0}^{T} \underbrace{-\mathbb{E}_{q(a_t, s_t|o_t)}\left[\ln p\left(o_t|s_t\right)\right]}_{\text{Accuracy}} + \underbrace{\mathbb{E}_{q(a_t|s_t)} D_{\mathrm{KL}}\left[q\left(s_t|o_t\right) \| p\left(s_t|s_{t-1}, a_{t-1}\right)\right]}_{\text{Complexity}} \\
&+ \underbrace{D_{\mathrm{KL}}\left[q\left(a_t|s_t\right) \| p\left(a_t|s_t\right)\right]}_{\text{Action Divergence}}
\end{aligned} \tag{56}
$$

This decomposition is useful since it allows for the interpretation of the VFE objective and, since it reveals all the distributions needed for the agent to conduct active inference. To remind the reader, the first term, accuracy (seen in equation 12), poses the minimisation of VFE as maximising the log-likelihood of expected observations, $\ln p(o|s)$. The first term therefore maximises the accuracy of the model in terms of new observations. The second term, complexity, acts as a regularisation term which keeps the variational posterior for states, $q(s|o)$, closer to the prior distribution of states (given previous states and actions, $p(s|s,a)$). Therefore, this restrains the minimisation of VFE from being limited to only maximum likelihood. Lastly, the action divergence term minimises the KL divergence between the variational action policy, $q(a|s)$, and prior action distribution, $p(a_t|s_t)$. This trains the variational action posterior on the action prior, which weights the action choices according to the agent's a-priori evaluation.

Turning to the deep learning approach for active inference, the above distributions can be approximated with parameterised deep neural networks [13]. Specifically, the likelihood $p(o|s)$, variational posterior for states $q(s|o)$, and the transition model $p(s_t|s_{t-1}, a_{t-1})$, are approximated using deep neural networks. The policy distribution, $q(a|s)$, is also approximated with a deep neural network. The approach of [13] can therefore be considered a policy gradient approach, one that approximates a stochastic policy function.

In contrast to our previous approach at performing the RL problem through inference [20], this active inference approach does not perform inference on a graphical model augmented with optimality nodes. Instead, this active inference agent encodes the rewards in equation 53. The rewards are embedded in the action prior, $p(a_t|s_t)$ via the prior preference distribution's appearance in the EFE objective function [13]. Instead of assuming a uniform prior, as with the control as inference approach, the action prior here adopts the discrete active inference approach. It is set equal to the softmax distribution of the EFE of future trajectories, $p\left(a_t|s_t\right) = \sigma\left(\beta \mathcal{G}_{a_{t:T}}\left(s_{t:T}, o_{t:T}\right)\right)$. The introduction of β, assists in defining an entropy hyperparameter which is similar to that used in the Boltzmann exploration approach in the exploration-exploitation dilemma section (see 3.4). Therefore, the action divergence term encodes an implicit exploratory behaviour, both via the EFE objective function and via the Boltzmann exploration hyperparameter. The use of the EFE value function to evaluate trajectories of the agent's experience, alongside the evaluation of a stochastic policy function, means that this active inference algorithm adopts an actor-critic model-free RL algorithm architecture.[49]

To evaluate the EFE, the algorithm which is called *deep active inference* [13], borrows from Q-learning

[49] One can review the actor-critic approach in section 3.2.

(equation 32). The limitation of tabular active inference agents, which evaluate every possible trajectory to estimate the EFE, is equivalent to the limitation of tabular RL agents, which calculate the expected cumulative reward for each trajectory. The use of the recursive Bellman equations allowed for a method of bootstrapping to calculate the expected cumulative reward in a method of approximate dynamic programming. Applying this principle to the EFE affords a similar recursive relationship. This gives the following expression for EFE as [1]:

$$\begin{aligned} \mathcal{G}_{a_{t:T}}\left(o_{t:T}, s_{t:T}\right) &= \sum_{t}^{T} \mathcal{G}_{t}\left(o_{t}, s_{t}\right) \\ \implies \mathcal{G}_{a_{t:T}}\left(o_{t:T}, s_{t:T}\right) &= \mathcal{G}_{t}\left(o_{t}, s_{t}\right) + \mathbb{E}_{p\left(o_{t+1}, s_{t+1} | s_{t}, a_{t}\right)}\left[\mathcal{G}\left(o_{t+1:T}, s_{t+1:T}\right)\right] \end{aligned} \tag{57}$$

Furthermore, the inclusion of the prior preference distribution, $\tilde{p}\left(o_{1:T}\right) = \prod_{t:T} \sigma\left(r\left(o_{t}\right)\right)$ serves to expand on the interpretation of EFE (previously presented in expression 13). As previously discussed, EFE can be decomposed into reward maximisation (an extrinsic motivation term) and an information gain term (an intrinsic motivation term). The information gain term acts as a *directed* exploratory motivation for the deep active inference agent. This provides the, aforementioned, benefit of efficiency, as compared to other random exploratory methods. As per the definition of EFE, the decomposition appears below. This time with the decomposition explicitly includes the reward maximisation [1]:

$$\begin{aligned} G_{t}\left(o_{t}, s_{t}\right) &= \mathbb{E}_{q\left(o_{t}, s_{t}\right)}\left[\ln q\left(s_{t}\right) - \ln \tilde{p}\left(o_{t}, s_{t}\right)\right] \\ &\approx -\mathbb{E}_{q\left(o_{t}, s_{t}\right)}\left[\ln \tilde{p}\left(o_{t}\right)\right] - \mathbb{E}_{q\left(o_{t}\right)} D_{\mathrm{KL}}\left[q\left(s_{t} | o_{t}\right) \| q\left(s_{t}\right)\right] \\ &\approx \underbrace{-\mathbb{E}_{q\left(o_{t}, s_{t}\right)}\left[r\left(o_{t}\right)\right]}_{\text{Reward Maximisation}} \underbrace{- \mathbb{E}_{q\left(o_{t}\right)} D_{\mathrm{KL}}\left[q\left(s_{t} | o_{t}\right) \| q\left(s_{t}\right)\right]}_{\text{Information Gain}} \end{aligned} \tag{58}$$

This decomposition allows for further interpretation of the action divergence term in the VFE objective, equation 56. This acts to minimise the divergence between the prior, *ideal* action distribution, $p(a_t|s_t)$, and the variational action policy, $q(a_t|s_t)$ (a deep neural network). This comes with the intuitive interpretation provided by the expansion of EFE, that the agent seeks to optimise its policy so as to match its action prior. The action prior, specifically, maximises both reward and information gain, via optimising the sum of EFE over time. As the policy prior takes a softmax form with the EFE as its argument, the agent therefore *desires* actions with probability proportional to the relative EFE path integral of each possible action.

To approximate EFE, the deep active inference approach borrows from the DQN toolkit [9]. The neural network used to approximate the EFE 'value function' is parameterised by ϕ. The network is defined to take, as input, the current state and observation pair and use these to predict the full EFE path integral value. The network parameters are trained using the squared error loss function between the current network and a network frozen from the past (a target) [13]. As with DQNs, the target network can be frozen for a fixed number of time-steps to stabilise the loss surface and remove correlation between the new EFE estimates and the target value function estimates. A random replay buffer can be used to store experience of the agent to reduce sequential correlation between observations used for training. In the line below, the bootstrapped EFE, predicted by the frozen network, is used in a recursive statement. The EFE value network is then [1, 13]:

$$\hat{\mathcal{G}}\left(o_{t:T} s_{t:T}\right) = \mathbb{E}_{q\left(o_{t}, s_{t}\right)}\left[r\left(o_{t}\right)\right] + \mathbb{E}_{q\left(o_{t}\right)} D_{\mathrm{KL}}\left[q\left(s_{t} | o_{t}\right) \| q\left(s_{t}\right)\right] + \mathbb{E}_{q\left(s_{t+1}, o_{t+1} | s_{t}, a_{t}\right) q\left(a_{t} | s_{t}\right)}\left[\mathcal{G}_{\phi}\left(o_{t+1:T}, s_{t+1:T}\right)\right] \tag{59}$$

The squared error loss function is defined for the EFE network as [1, 13]:

$$|\hat{G}\left(o_{t}, s_{t}\right) - G_{\phi}\left(o_{t}, s_{t}\right)|^{2} \tag{60}$$

Looking back at the VFE expansion in expression 56, it should now be clear how all the distributions in the expression are obtained. Having trained a network for EFE, one can then obtain the prior action distribution, $p(a_t|s_t)$, from the softmax function. Thereafter, a policy gradient approach can be used to train the policy network $q(a|s)$ on the minimisation of VFE objective function. $q(a_t|s_t)$ is a neural network representing the policy network. The transition dynamics, $p(s_t|s_{t-1}, a_{t-1})$, and distributions $q(s_t|o_t)$ and $p(o_t|s_t)$ are also obtained from neural networks trained on the agent's experience of the environment. Using these distributions, the VFE objective function can be evaluated and its gradients can be computed for the purpose of stochastic gradient descent [1].

The complete algorithm representing deep active inference appears below. As the algorithm is inspired by a deep actor-critic model-free RL architecture, it thus effectively scales the active inference agent to larger state-spaces and longer time horizons. Interestingly, this method was found to be performant on some proof-of-concept benchmark tasks from OpenAI Gym [23] in [13]. Next, we study an algorithm inspired by model-based RL methods [14].

Algorithm 4 Deep Active Inference [13]

Initialisation:

Parameters θ, ϕ, ξ, Ψ

Observation networks $q_\theta(s|o), p_\theta(o|s)$

State transition network $p_\phi(s_t|s_{t-1}, a_{t-1})$

Policy network $q_\xi(a|s)$

Bootstrapped EFE network $\mathcal{G}_\Psi$

Receive prior state s_0

Take prior action a_0

Receive initial observation o_1

Receive initial reward r_1

function ACTION-PERCEPTION LOOP(θ, ϕ, ξ, Ψ)

while $t < T$ **do**

$\hat{s}_t \leftarrow q_\theta(s \mid o)\,(o_t)$ ▷ Infer the expected state from the observation.

$\widehat{s_{t+1}} \leftarrow p_\phi\,(s \mid s_{t-1}, a_{t-1})\,(\hat{s})$ ▷ Predict the state distribution for the next time-step.

$a_t \sim q_\xi(a \mid s)$ ▷ Sample an action from the policy and execute it.

Receive observation o_{t+1}

Receive reward r_{t+1}

$\hat{s}_{t+1} \leftarrow q_\theta(s \mid o)\,(o_{t+1})$ ▷ Infer expected state from the next observation.

Compute the bootstrapped EFE estimate from the current state and action.

$\widehat{G(s,a)} \leftarrow r_{t+1} + E_{q(s_{t+1})}\left[\ln \widehat{s_{t+1}} - \ln \hat{s}_{t+1}\right] + E_{q(s_{t+1},a_{t+1})}\left[G_\psi\left(s_{t+2}, a_{t+2}\right)\right]$

Compute the VFE

$F \leftarrow E_{Q(s)}[\ln p(o \mid s)] + KL\left[s_{t+1} || \widehat{s_{t+1}}\right] + E_{Q(s)}\left[\int da Q_\xi(a \mid s)\sigma\left(-\beta G_\psi(s,a)\left(s_{t+1}\right)\right) + \mathcal{H}\left(Q_\xi(a \mid s)\right)\right]$

$\theta \leftarrow \theta + \alpha \frac{dF}{d\theta}$

$\phi \leftarrow \phi + \alpha \frac{dF}{d\phi}$

$\xi \leftarrow \xi + \alpha \frac{dF}{d\xi}$

$L \leftarrow \left\| \widehat{G(s,a)} - G_\psi(s,a) \right\|^2$

$\psi \leftarrow \psi + \alpha \frac{dL}{d\psi}$

end while

Return Policy $q_\xi(a|s)$

end function

4.2 A Model-Based RL Approach to Active Inference

Examining the model-free approach taken above, it is possible to replace the bootstrapped EFE recursive statement with a model-based expectation by using the generative model of the agent to directly estimate trajectories (and the EFE) via a model-based planning approach [1, 14]. Even though this may incur greater computational cost, this approach is adopted in [14].

The model-based approach takes aim at the generative model proposed by the tabular active inference framework. Recall that in this (tabular) context, the agent held a generative model which it used to make predictions and conduct active inference via evaluation (of the EFE) and, *planning*. In this (tabular) case, the agent's generative model was used for planning and the evaluation of every possible policy at every time-step. For this reason tabular active inference can be considered a model-based planning algorithm that employs exhaustive planning in the state-action space [1]. Once again, when turning to the problem of scaling the tabular active inference approaches, the use of deep neural networks is key but this time we review the approach in a model-based setting.

As with the model-free approach the main distributions appearing in the POMDP model are parameterised by deep neural networks [1, 14]. Specifically, the transition model, $p(s_t|s_{t-1}, a_{t-1})$, and the likelihood function, $p(o|s)$, are approximated by deep neural networks. However, the variational distributions in, $q(s, o, \theta)$, now become the outcome of a belief evaluation scheme. This takes expectations of the true generative model, given the current time variational distribution. Primarily, the difference between the model-free and model-based approach is that the stochastic policy function, $q(a_t|s_t)$, is no longer approximated by a policy neural network.[50] Instead, the policy function is now treated as the output of model-based planning. This means that the optimal policy becomes a softmax distribution (of a negative free energy functional). This softmax distribution minimises a new objective functional, the *free energy of the expected future* [14]. The free energy of expected future path integral is approximated by performing MC sampling from the agent's trained generative model (i.e. model-based planning). The most likely policy in the resulting softmax policy distribution is that which minimises a free energy function (which is similar to EFE).

The derivation below establishes the optimal policy as a softmax distribution (of a free energy path integral, $\tilde{\mathcal{F}}_\pi$) [14]. This softmax distribution is optimal as it minimises the free energy of the expected future (FEEF) objective function, $\tilde{\mathcal{F}}$.[51] Previously, the active inference agent desired to minimise its VFE, $\mathcal{F}$, which was the KL divergence between its biased generative model, $\tilde{p}$, and its variational predictive distribution, q. The agent's EFE, $\mathcal{G}$, was the expectation of the VFE quantity with the expectation taken using the current time variational distribution, q. In contrast to these, the FEEF objective function, $\tilde{\mathcal{F}}$, includes a belief over policies, $\pi = \{a_o, a_1, ..., a_T\}$ which are themselves random variables. Where VFE was defined for a single time point, t now the FEEF objective is defined for π, which refers to a temporal sequence of actions. Hence the augmentation of the VFE quantity with inference as to the policy π means that the FEEF objective measures the divergence between a sequence of beliefs and the agent's biased generative model. The FEEF is defined as:

[50] trained using the KL divergence between itself and the action prior which was represented by the softmax of the EFE value function, in expression 56.

[51] Note that the same distributions that were used in the model-free deep active inference section, appear once more here.

$$
\begin{aligned}
\tilde{\mathcal{F}} &= D_{\mathrm{KL}}\left(q\left(o_t, s_t, \theta, \pi\right) \| \tilde{p}\left(o_t, s_t, \theta\right)\right) \\
&= \mathbb{E}_{q(o_t, s_t, \theta, \pi)}\left[\ln q\left(o_t, s_t, \theta \mid \pi\right) + \ln q(\pi) - \ln \tilde{p}\left(o_t, s_t, \theta, \pi\right)\right] \\
&= \mathbb{E}_{q(\pi)}\left[\mathbb{E}_{q(o_t, s_t, \theta \mid \pi)}\left[\ln q(\pi) - \left[\ln \tilde{p}\left(o_t, s_t, \theta\right) - \ln q\left(o_t, s_t, \theta \mid \pi\right)\right]\right]\right] \\
&= \mathbb{E}_{q(\pi)}\left[\ln q(\pi) - \mathbb{E}_{q(o_t, s_t, \theta \mid \pi)}\left[\ln \tilde{p}\left(o_t, s_t, \theta\right) - \ln q\left(o_t, s_t, \theta \mid \pi\right)\right]\right] \\
&= \mathbb{E}_{q(\pi)}\left[\ln q(\pi) - \left[-\mathbb{E}_{q(o_t, s_t, \theta \mid \pi)}\left[\ln q\left(o_t, s_t, \theta \mid \pi\right) - \ln \tilde{p}\left(o_t, s_t, \theta\right)\right]\right]\right] \\
&= \mathbb{E}_{q(\pi)}\left[\ln q(\pi) - \ln e^{-\left[-\mathbb{E}_{q(o_t, s_t, \theta \mid \pi)}\left[\ln q(o_t, s_t, \theta \mid \pi) - \ln \tilde{p}(o_t, s_t, \theta)\right]\right]}\right] \\
&= \mathbb{E}_{q(\pi)}\left[\ln q(\pi) - \ln e^{-D_{\mathrm{KL}}(q(o_t, s_t, \theta \mid \pi) \| \tilde{p}(o_t, s_t, \theta)}\right)\right] \\
&= D_{\mathrm{KL}}\left(q(\pi) \| e^{-D_{\mathrm{KL}}(q(o_t, s_t, \theta \mid \pi) \| \tilde{p}(o_t, s_t, \theta)}\right) \\
&= D_{\mathrm{KL}}\left(q(\pi) \| e^{-\tilde{\mathcal{F}}_\pi}\right)
\end{aligned}
\tag{61}
$$

This means that a KL divergence of zero gives [14]:

$$
\tilde{\mathcal{F}} = 0 \Rightarrow D_{\mathrm{KL}}\left(q(\pi) \| \left(e^{-\tilde{\mathcal{F}}_\pi}\right)\right) = 0 \tag{62}
$$

where (given π):

$$
\tilde{\mathcal{F}}_\pi = D_{\mathrm{KL}}\left[q\left(o_t, s_t, \theta \mid \pi\right) \| \tilde{p}\left(o_t, s_t, \theta\right)\right] \tag{63}
$$

Thus, the FEEF is minimised when $q(\pi) = \sigma\left(-\tilde{\mathcal{F}}_\pi\right)$.[52] Hence, policy selection arises from the identification of the stochastic policy $q(\pi)$, that minimises $\tilde{\mathcal{F}}$ and policies are more likely when they minimise $\tilde{\mathcal{F}}_\pi$. As per our usual approach, the minimisation of the expression $\tilde{\mathcal{F}}_\pi$, can be expanded for interpretation [1]:

$$
\begin{aligned}
\tilde{\mathcal{F}}_\pi &= D_{\mathrm{KL}}\left[q\left(o_t, s_t, \theta \mid \pi\right) \| \tilde{p}\left(o_t, s_t, \theta\right)\right. \\
&\approx \underbrace{\mathbb{E}_{q(s_t \mid \pi)} D_{\mathrm{KL}}\left[q\left(o_t \mid s_t\right) \| \tilde{p}\left(o_t\right)\right]}_{\text{Extrinsic Value}} - \underbrace{\mathbb{E}_{q(o_t; \theta)} D_{\mathrm{KL}}\left[q\left(s_t \mid o_t, \theta\right) \| q\left(s_t\right)\right]}_{\text{State Information Gain}} \\
&- \underbrace{D_{\mathrm{KL}}\left[q\left(\theta \mid s_t\right) \| q(\theta)\right]}_{\text{Parameter Information Gain}}
\end{aligned}
\tag{64}
$$

The first of the three terms is called extrinsic value. This gives a measurement defining by how much the expected observations diverge from the desired observations (given by the prior preference distribution $\tilde{p}(o_t)$). The extrinsic value term presents a divergence objective. This means that the agent seeks to match its predictive distribution (of observations) to its distribution of preferences. This encodes the reward maximisation component (that is, if the distribution of preferences does encode reward, as in equation 53). In [14] the authors encode their biased generative model as: $p(o, s, \theta) = p(s, \theta|o)\tilde{p}(o)$. This treats rewards as a separate observation mode and encodes $\tilde{p}(o)$ as a distribution over preferred rewards. This implies that the biased generative model is only biased with respect to observing greater amounts of reward (as with RL agents).

The second and third terms encode the information seeking objectives for states and parameters respectively. The state information gain term is the same as that appearing in the model-free setting in equation 58.[53] For the case of states this term is associated with the perceptive inference of the agent. For the case

[52] Here σ once again represents the softmax distribution of its argument.

[53] The meaning of the information gain term was reviewed for equations 13 and 58.

of parameters this is associated with the model learning of the agent, thus explicitly demonstrating the unification of perception, action and learning under a unified objective function, as with our active inference formulation in section 2.3.

Another key difference between the model-based and the model-free settings is that, for the model-based setting, the parameters are reasoned about in a Bayesian manner (as can be seen in the generative model in expression 61). Once again we mention that the key benefit of the framing of the objective function in this manner is that it induces *goal-directed exploration* since the information gain terms furnish the objective function with exploratory objectives that reward the gaining of information on the posterior of states and parameters. This approach also offers a mathematically principled origin for the goal-directed exploration of states and parameters [1]. The joint optimisation of the exploratory and exploitatory objectives means that the agent is not simply rewarded for reducing uncertainty via random exploration but is explicitly rewarded for reducing uncertainty about states and actions that will be rewarding for the agent. This is the key benefit that the active inference approach provides contemporary RL methods [14].

As the transition model's parameters, θ, are included in the generative model, they are therefore reasoned about in the same way as states and observations. This manifests itself in the FEEF objective function with the parameter information gain term. In [14], the approximate distribution over the parameters, $q(\theta)$, is formulated as a mixture model of distributions of parameters. These parameters result from an ensemble of transition models $p(s|s,\theta,\pi)$ (which are modelled with neural networks). Each transition model has independently initialised parameter distributions, $p(\theta)$. The transition models are trained on different samples from an experience replay buffer (see DQN, section 3.2). This approach allows for a multimodal posterior distribution over parameters, $q(\theta|s_t)$ and, has been shown to offers superior representation of the true parameter posterior and to avoid overfitting in sparse data settings [1]. This is another benefit of the model-based approach taken in [14].

For the next modelling component, a reward model can be learned from the agent's experienced interactions with the environment. This can then be used for model-based planning which occurs through simulated sample trajectories from our model. The reward model can be approximated by a neural network and trained using data from a random replay buffer. The FEEF objective function defines the extrinsic reward as the KL divergence between the prior preference distributions for observations and the likelihood function. In [14], the reward model replaces the likelihood function, $p(o|s)$, as their environment is an MDP, meaning that the only observation is the reward. This is trained using the extrinsic value objective function.

The implementation of the FEEF objective function in [14] optimises the policy, $q(\pi)$ at each time-step, executing the first action specified by the most likely policy. This requires three components. First, a method for evaluating beliefs about future states and observations, $q(s,o,\theta|\pi)$. Second, a method for evaluating the FEEF objective function and, thirdly a method for optimising $q(\pi) = \sigma(-\mathcal{F}_\pi)$. For evaluating beliefs, the generative model is, once again, factorised using the Markov property as [14]:

$$q\left(\mathrm{s}_{t:T}, \mathrm{o}_{t:T}, \theta \mid \pi\right) = q(\theta) \prod_{t}^{T} q\left(\mathrm{o}_t \mid \mathrm{s}_t, \theta, \pi\right) q\left(\mathrm{s}_t \mid \mathrm{s}_{t-1}, \theta, \pi\right) \tag{65}$$

Future beliefs are evaluated as (a model-based planning approach due to the expectations being taken using samples from the current-time predictive distribution under a particular policy) [14]:

$$\begin{aligned} q\left(o_t \mid s_t, \theta, \pi\right) &= \mathbb{E}_{q(s_t|\theta,\pi)}\left[p\left(o_t \mid s_t\right)\right] \\ q\left(s_t \mid s_{t-1}, \theta, \pi\right) &= \mathbb{E}_{q(s_{t-1}|\theta,\pi)}\left[p\left(s_t \mid s_{t-1}, \theta, \pi\right)\right] \end{aligned} \tag{66}$$

A planning horizon H is used for the evaluation of the FEEF objective such that $-\tilde{\mathcal{F}}_\pi = \sum_t^{t+H} -\tilde{\mathcal{F}}_{\pi_t}$.[54]

[54] ... thus applying the ergodicity assumption from section 2.4.

This is computed using the KL divergence for the extrinsic value (reward) term in equation 64. For the purpose of evaluation of the information gain terms, the algorithm reformulates information gain in terms of entropy. This can be seen below, for the case of information gain in the transition model parameters. This decomposition is interpreted as maximising the entropy of the average state minus, the average of the entropy. Due to the adoption of the ensemble approach to modelling the transition dynamics distributions, these entropy terms can be easily computed [14].

$$\begin{aligned}
&\mathbb{E}_{q(s_t|\theta)} D_{\mathrm{KL}}\left(q\left(\theta \mid s_t\right) \| q(\theta)\right) \\
&= \mathbb{E}_{q(s_t|\theta)q(\theta|rs)}\left[\ln q\left(\theta \mid s_t\right) - \ln q(\theta)\right] \\
&= \mathbb{E}_{q(s_t,\theta)}\left[\ln q\left(s_t \mid \theta\right) + \ln q(\theta) - \ln q\left(s_t\right) - \ln q(\theta)\right] \\
&= \mathbb{E}_{q(s_t,\theta)}\left[\ln q\left(s_t \mid \theta\right) - \ln q\left(s_t\right)\right] \\
&= \mathbb{E}_{q(\theta)q(s_t|\theta)}\left[\ln q\left(s_t \mid \theta\right)\right] - \mathbb{E}_{q(\theta)q(s_t|\theta)}\left[\ln \mathbb{E}_{q(\theta)} q\left(s_t \mid \theta\right)\right] \\
&= -\mathbb{E}_{q(\theta)}\left[\mathcal{H}\left[q\left(s_t \mid \theta\right)\right]\right] + \mathcal{H}\left[\mathbb{E}_{q(\theta)} q\left(s_t \mid \theta\right)\right]
\end{aligned} \tag{67}$$

Putting these components together yields the algorithm below. Note that this approach, from [14], uses a cross-entropy method (CEM) planner which was discussed in section 3.2.1. Once again a 'proof-of-concept' demonstration, in the sparse reward OpenAI Gym environment called mountain car [23], resulted in the FEEF algorithm being performant (as compared to some benchmark algorithms).[55]

4.3 Model-Free or Model-Based? - A Hybridised Approach

Having successfully tackled the problem of scaling active inference agents, via the study of both a model-free and a model-based RL-inspired approach, it is important to review the benefits and shortcomings of these. Instead of approaching this from just the perspective of model-free and model-based RL, the authors of [21] construct new taxonomic categories for RL algorithms. This is the context in which we review the benefits of the model-free and model-based approaches and in which we study the construction of a hybridised version of these algorithms, for scaling the active inference agent using deep RL methods.

The new taxonomic categories that [21] discuss consist of two orthogonal dimensions. The first is the familiar dimension of whether a single action is inferred by the RL agent or, whether (model-based) planning takes place. Some "single-action" algorithms in the RL field include Q-learning (DQN) and policy gradient methods (REINFORCE). These are the typical model-free algorithms which look to map states to actions. Within the planning category, the authors of [21] provide some examples of model-based planners used in RL.

The benefits associated with model-free algorithms are that of increased asymptotic performance as compared to their model-based counterparts, whose model allows them to generalise their knowledge to new tasks and, which are more sample efficient and able to learn from fewer trials [45]. The second orthogonal dimension of the new RL taxonomy in [21] is oriented around two types of inference.

4.3.1 Iterative vs Amortised Inference

The two types of inference mentioned in [21] are *iterative* and *amortised* inference. In terms of the problem of variational Bayesian inference, iterative inference seeks to directly optimise the parameters, θ, of the variational posterior in order to minimise the KL divergence between the true and variational posteriors (VFE). According to [21], this process is most often carried out iteratively for each data point, although could theoretically be carried out over a single step. Framing the RL problem as one of inference as

[55] Mountain car is discussed in section 3.4.

Algorithm 5 Free Energy of Expected Future [14]

Initialisation:

Planning horizon length H

Optimisation iterations I

Number of candidate policies J

Receive prior state s_0

Likelihood function $p(o_\tau, s_\tau)$

Transition distribution $p\left(s_\tau \mid s_{\tau-1}, \theta, \pi\right)$

Parameter distribution $p(\theta)$

Global prior observation distribution $\tilde{p}(o_\tau)$

Factorised belief over action sequences $q(\pi) \leftarrow N(0, \mathbb{I})$

for Optimisation iteration $i = 1 \dots I$ **do**
  Sample J candidate policies from $q(\pi)$
  for candidate policy $j = 1 \dots J$ **do**
    $\pi^{(j)} \sim q(\pi)$
    $-\tilde{\mathcal{F}}_\pi^j = 0$
    for $\tau = t \dots t + H$ **do**
      $q\left(s_\tau \mid s_{\tau-1}, \theta, \pi^{(j)}\right) = \mathbb{E}_{q\left(s_{\tau-1} \mid \theta, \pi^{(j)}\right)}\left[p\left(s_\tau \mid s_{\tau-1}, \theta, \pi^{(j)}\right)\right]$ ▷ Belief evaluation
      $q\left(o_\tau \mid s_\tau, \theta, \pi^{(j)}\right) = \mathbb{E}_{q\left(s_\tau \mid \theta, \pi^{(j)}\right)}\left[p\left(o_\tau \mid s_\tau\right)\right]$
      $-\tilde{\mathcal{F}}_\pi^j \leftarrow -\tilde{\mathcal{F}}_\pi^j + E_{q\left(s_\tau, \theta \mid \pi^{(j)}\right)}\left[D_{\mathrm{KL}}\left(q\left(o_\tau \mid s_\tau, \theta, \pi^{(j)}\right) \| \tilde{p}\left(o_\tau\right)\right)\right] + \mathcal{H}\left[q\left(s_\tau \mid s_{\tau-1}, \theta, \pi^{(j)}\right)\right] +$
        $-\mathbb{E}_{q(\theta)}\left[\mathcal{H}\left[q\left(s_\tau \mid s_{\tau-1}, \pi^{(j)}, \theta\right)\right]\right]$ ▷ Evaluating $\mathcal{F}_\pi$
    end for
  end for
  $q(\pi) \leftarrow \sigma\left(-\tilde{\mathcal{F}}_\pi^j\right)$ ▷ Optimising the policy distribution, fitting a softmax
end for
Return $q(\pi)$

seen in section 3.3 means that one can frame the variational posterior as being a posterior over actions. Hence, framing model-based planning algorithms as updating posterior parameters for posteriors of action sequences, casts several model-based planning algorithms as a process of iterative inference [21].

In contrast, amortised variational inference learns the parameters of the variational distribution *via* a function, f_ϕ, which maps data to the parameters of the variational distribution: $\theta = f_\phi(\text{data})$ [21, 46]. Amortised inference optimises the parameters ϕ such that the function, f_ϕ, outputs the optimised parameters, θ, that minimise the KL divergence between the true and approximate posteriors (VFE). A frequent implementation of this amortisation function is via a neural network with parameters ϕ. This can be trained via gradient descent on the VFE objective function. In the context of the control as inference section (section 3.3), where we described the posterior in terms of actions, the parameterised function, f_ϕ, corresponds to a policy [21]. An example of this can be seen in equation 36. The taxonomy from [21] can be viewed in figure 13.

It may be helpful to mathematically describe the subtle difference between iterative and amortised inference before discussing the benefits of either approach. Suppose that the variational distribution is Gaussian with parameters $\theta = \{\mu, \sigma\}$, then iterative inference will try to optimise these *directly*, to fit some data. Iterative inference can be described by the following minimisation (in terms of a posterior over states) [1]:

$$\underset{\theta}{\operatorname{argmin}}\ \mathbb{E}_{q(s;\theta)}[\ln q(s;\theta) - \ln p(o, s)] \tag{68}$$

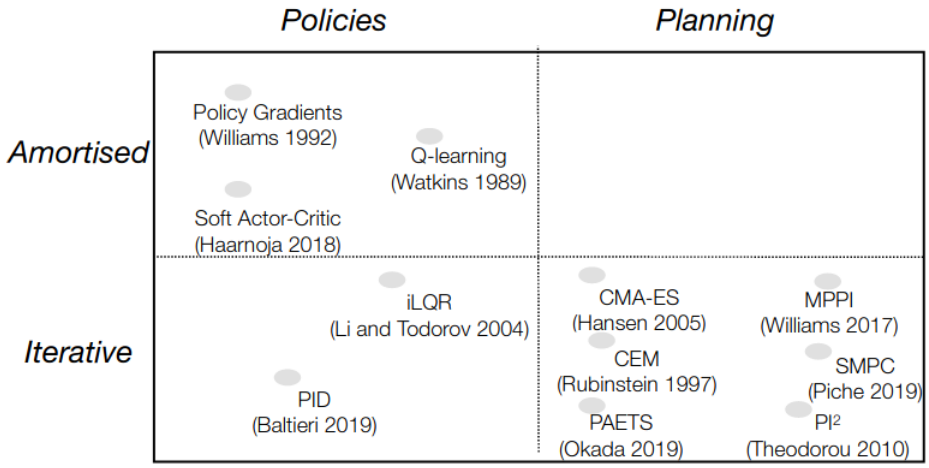

Figure 13: The taxonomy scheme provided in [21]. In this scheme the label 'policies' refers to inference about a single action (for instance from a state mapping or a value function). Model-free RL algorithms appear in the top left, corresponding to amortised policies. Model-based planners appear in the iterative planning quadrant and, control theory infers iterative policies. The upper right quadrant leaves potential room for novel algorithms.

Amortised inference is described by the following (indirect optimisation via the amortisation function f_ϕ), where D represents available data:[56]

$$\underset{\phi}{\operatorname{argmin}}\ \mathbb{E}_{p(D)}\left[\mathbb{E}_{q\left(s;\hat{\theta}=f_\phi(D)\right)}\left[\ln q\left(s;\hat{\theta}=f_\phi(D)\right)-\ln p(o,s)\right]\right] \tag{69}$$

In the case of amortised inference, the amortisation function requires optimisation across a whole dataset. This means that amortised methods are required to generalise in a way that iterative inference methods do not have to. However, once learned the amortisation function can be used to efficiently estimate any variational parameters $\hat{\theta}$ for any data point. For instance, in the case of an action posterior, the function may output an action for a given state or observation.

Thinking in terms of an MDP (and a CEM planner) and considering a new state that the agent arrives at, the iterative inference approach can start from scratch at each of these new states and optimise the variational posterior over actions afresh (via iterative evaluated roll-outs from the planner). An example of this is the case of model-based planning with the FEEF objective function. Alternatively, in terms of a CEM model-based planner (section 3.2.1), [21] views the agent optimising its policy posterior over action sequences, which are optimised over the course of multiple iterations, as iterative inference.

Turning to the benefits of either approach, we note that amortised inference offers the benefit of inferring the parameters quickly once learned. For a neural network this takes one feed-forward pass. Furthermore the model learning is split over the entire dataset. The use of the whole dataset *amortises* (pays-off) the cost of inference across the whole dataset. Iterative inference offers the ability to learn on fewer data-points (for instance with the sample efficiency of model-based RL) but, incurs the computational cost of performing gradient descent for every inference that the system wishes to make. One issue with amortised methods is the requirement of the amortisation function to generalise across the dataset. This is not necessary for iterative methods and, can cause variational parameters found by amortised methods to be worse estimates than those of iterative inference [1].

[56] which is context dependent.

4.3.2 Control as Hybrid Inference

Possible advantages may be gained from combining the use of both amortised and iterative inference in a hybridised approach, especially to ameliorate the disadvantages of both [21, 46]. This approach is taken by the authors of [17] and offers a third algorithm for scaling active inference that is inspired by the use of deep neural networks in RL algorithms and by a particular choice of combination of model-free and model-based methods.

Model-free RL methods share the properties of amortised inference as they require a whole dataset to train. They also require a wide range of experience for good generalisation. However, once trained, model-free RL methods can quickly compute (actions) for any state. Model-based RL methods share properties with iterative inference as they are sample efficient but can be more computationally expensive per data-point due to use of iterative planning for each state. Despite these commonalities and despite the variational parameters of amortised methods often providing worse estimates [1], model-free approaches achieve higher asymptotic performance than model-based methods [45]. The reason for this may be the second, orthogonal dimension of the taxonomy of [21] (figure 13), where inferring action plans adds additional complexity than inferring single actions.

The algorithm provided by [17] looks to take the high asymptotic performance and rapid inference of amortised methods and combine this with the sample efficiency of iterative inference methods. Specifically, the authors use a model-free policy as the starting point for a model-based planner, such as the CEM. The model-free method can be trained by minimising a VFE functional (similarly to algorithm 4) and, the model-based planner can optimise the policy posterior by evaluating which action sequences minimises VFE the most (similarly to algorithm 5). The combination of these, therefore, provides the implementation of another scaled active inference inference agent.

The algorithm begins by initialising a transition model neural network, $p_\lambda(s_{t+1}|s_t, a_t)$. To obtain the initial action distribution (for the planner) from the model-free policy, the transition model is used to provide the "next state" subsequent to a state-action pair. The transition model in [17] was trained using maximum likelihood for the parameter λ. This was trained using an ensemble approach on data gathered in a random experience-replay buffer (see DQN, section 3.2).

The amortised component of the algorithm initialises a model-free policy $q_\phi(a_t|s_t;\theta)$ and, the amortisation function $\theta = f_\phi(s_t)$. As with the FEEF algorithm, the initial action distribution for a model-based planning algorithm can be provided by a Gaussian distribution, with a mean of zero and, variance set as a hyperparameter [1]. Instead, the model-free policy and amortised inference approach provide the mean and variance parameters, $\theta = \{\mu, \sigma^2\}$, for an initial action (Gaussian) distribution, $q_\phi(a_t|s_t;\theta) = N(\theta_t)$. θ is obtained via the amortisation function $\theta = f_\phi(s_t)$, where the parameters, ϕ, are obtained via the minimisation of VFE. The amortisation function is trained on data from the random experience-replay buffer. The resulting policy in [17] uses a soft actor-critic [47] (see section 3.2) which is similar to the policy emerging from the FEEF algorithm, algorithm 5.

As mentioned, the model-free policy gives an initial action distribution for the model-based planning algorithm, which comprises the iterative inference component of the hybridised algorithm. The iterative component optimises a policy posterior using evaluations of state-action roll-outs from the model-based planner (for a fixed time horizon, H). The initial distribution (for the planner) is obtained using the trained transition model and, given an initial state distribution $p(s_t) = \delta(s_t)$ [17]:

$$\delta\left(s_t\right) \prod_{t'=t}^{T} p_\lambda\left(s_{t'+1}|s_{t'}, a_{t'}\right) q_\phi\left(a_{t'}|s_{t'};\theta\right) \tag{70}$$

From this, the initial action distribution over sequences of actions $q_\phi(a_{t:T}|s_{t:T};\theta)$, can be recovered, for

the parameters $\theta = \{(\mu_t, \sigma_t^2)\}_t^T$. Using this initial distribution of actions the agent observes, at each time-step, a state. The action posterior can then be iteratively updated (via planning) in order to minimise VFE. The iterative inference procedure used in [17] uses the iterative update for the posterior, below. This is known as mirror descent. Here, $\mathcal{W}$ is the expected cumulative reward such that $\mathcal{W}(a_{t:T}) = \mathbb{E}_{q(s_{t:T}|a_{t:T},s_t;\theta)}[C_T(\{r(s_t, a_t)\}_{t=t}^T)]$ [17].[57]

$$q^{(i+1)}\left(a_{t:T};\theta\right) \leftarrow \frac{q^{(i)}\left(a_{t:T};\theta\right) \cdot \mathcal{W}\left(a_{t:T}\right) \cdot q^{(i)}\left(a_{t:T};\theta\right)}{\mathbb{E}_{q^{(i)}(a_{t:T};\theta)}\left[\mathcal{W}\left(a_{t:T}\right) \cdot q^{(i)}\left(a_{t:T};\theta\right)\right]} \tag{71}$$

This mirror descent procedure, as illustrated by [48], provides an effective means for updating the posterior. Recent work by [48] has demonstrated that this mirror descent algorithm provides a Bayesian generalisation for the cross-entropy method (section 3.2.1) and also for a number of other stochastic optimisation methods used in model-based planning [21, 17]. The procedure gives more weight, in the policy posterior, to the evaluated action sequences that achieve greater amounts of cumulative reward (according to the current model). After a fixed number of updates the algorithm returns the posterior policy distribution. This produces the overall algorithm below. The algorithm is able to retain the sample efficiency of model-based RL, while obtaining the asymptotic performance of model-free RL methods [17]. Furthermore, the initial guess for the model-based planning component significantly reduces the search space. Combining this property with the use of a value function in an amortised inference approach may offer the benefit of fast value estimates beyond the planning horizon. The authors also note that since the certainty of the amortised predictions increase over the course of training, the possibility of terminating the iterative inference once a particular threshold is reached may decrease the computational costs of the method. The authors ran a proof-of-concept test of the algorithm in the OpenAI Gym environment, called Half-Cheetah [23]. Once again this demonstrated the performance of a scaled active inference algorithm against some benchmarks.

[57] C_t is defined in equation 21.

Algorithm 6 Control as Hybrid Inference [17]

Initialisation:

Planning horizon length H

Optimisation iterations I

Number of candidate policies J

Receive prior state s_t

Transition distribution $p_\lambda\left(s_t \mid s_{t-1}, a_{t-1}\right)$

Amortisation function $f_\phi(s_t)$

Parameters ϕ, λ

Replay Buffer $\mathcal{B}$

Train $p_\lambda(s_{t+1}|s_t, a_t)$ with data from $\mathcal{B}$

function AMORTISED INFERENCE $(\theta = f_\phi(s_t))$

Sample from $\mathcal{B}$

Obtain ϕ that minimises VFE: $\mathbb{E}_{q(\{(s_t,a_t)_t^T\})}\left[\sum_{t=1}^{T} r\left(s_t, a_t\right)\right] + \mathcal{H}\left[q_\theta\left(a_{1:T} \mid s_{1:T}\right)\right]$

$p_\phi := \delta\left(s_t\right) \prod_{t'=t}^{T} p_\lambda\left(s_{t'+1} \mid s_{t'}, a_{t'}\right) q_\phi\left(a_{t'} \mid s_{t'}; \theta\right)$

Extract $\theta^{(1)} = \left\{\mu_{t:T}, \sigma_{t:T}^2\right\}$ from p_ϕ

Initialise $q\left(a_{t:T}; \theta\right)$ with parameters $\theta^{(1)}$

end function

function ITERATIVE INFERENCE $(q\left(a_{t:T}; \theta\right))$

for Optimisation iteration $i = 1 \ldots I$ **do**

Sample J candidate policies from $\left\{\left(a_{t:T}\right)_j \sim q\left(a_{t:T}; \theta\right)\right\}_{j=1}^{J}$

Initialise weights $\mathbf{W}^{(i)} := \left\{w_j^{(i)}\right\}_{j=1}^{J}$

for Candidate policy $j = 1 \ldots J$ **do**

$w_j^{(i+1)} \leftarrow \frac{\mathcal{W}\left(\left(a_{t:T}\right)_j\right) \cdot q^{(i)}\left(\left(a_{t:T}\right)_j; \theta\right)}{\sum_{j=1}^{J}\left[\mathcal{W}\left(\left(a_{t:T}\right)_j\right) \cdot q^{(i)}\left(\left(a_{t:T}\right)_j; \theta\right)\right]}$

$\theta^{(i+1)} \leftarrow$ refit $\left(\mathbf{W}^{(i+1)}\right)$

end for

end for

end function

Return $q(a_{t:T}, \theta)$

5 Conclusions and Discussion

Having approached the problem of scaling tabular active inference approaches using the insights and successes [9, 35, 36] of model-free [9, 10, 11] and model-based [8, 12] deep RL, we have reviewed a means through which the active inference agent and its approach to sequential decision-making in an uncertain environment can be successfully scaled, via the discussion of three contemporary algorithms [13, 14, 17]. Drawing contrast to tabular active inference, the discussed, contemporary research makes use of deep neural networks to model some of the key distributions of the active inference agent's VFE objective function. Methods from model-free and model-based deep RL inspire two particular deep active inference algorithms [13, 14]. A third algorithm takes a hybridised approach [17].

One design choice likens the EFE path integral to a value function in deep model-free RL [13]. This can be approximated using a recursive Bellman expression which includes a bootstrapped estimate of the value function from the previous time-step, and thus follows a temporal difference approach [8]. Subsequently, an action posterior network can be trained to minimise the KL divergence between itself and the softmax of the EFE value function. Overall, this presents a model-free, actor-critic styled algorithm in [13].

As the EFE path integral can be expressed using a recursive expression and, since it can be separated into independent contributions from each time-step [1], a second algorithm [14] is able to easily enumerate the EFE path integral, in terms of sample trajectories from a model-based planner. In [14], this approach is taken using a slightly modified free energy objective. This model-based algorithm makes use of the state transition model to sample states, using the current-time policy. Then, model-based planning at each time-step can be used to derive an optimal and flexible policy. As with model-based RL and, as compared to the model-free active inference algorithm, this presents the advantage of sample efficiency for the agent.

Thirdly, a hybridisation of the model-free and model-based algorithms unites the advantages of both [17]. As we can understand the problem of action selection as a problem of inference, we can interpret the hybrid as combining two types inference. These are iterative inference and amortised inference [21]. Iterative inference functions by directly and iteratively optimising the parameters of a variational distribution. Amortised inference operates by making use of an optimised, parameterised amortisation function. This maps observations to the parameters of a variational distribution. These two types of inference categorise algorithms in a novel, two-dimensional taxonomy for RL algorithms, from [21]. This taxonomy highlights the area of *amortised inference plans* as one requiring future research [1, 21].

The authors of [17] combine the iterative and amortised inference schemes via the hybridisation of the model-free and model-based deep active inference algorithms. Here, an amortised model-free policy is used to initialise the model-based planning algorithm. This combines the sample efficiency of model-based algorithms with the asymptotic and computational efficiency of model-free algorithms. This choice of hybrid, however, remains a design choice, while other alternatives are available.[58]

Thus, active inference agents have found benefit in taking a deep RL approach. As both active inference and RL agents can be seen to address the same problem (of agent-based decision-making in an uncertain environment to achieve some goal) it may be useful to ask the question: how do active inference agents differ from RL agents? In this work we have studied, via derivation and decomposition, that the active inference agent's objective function has something to offer the RL agent. Specifically, the directed intrinsic exploratory motivation, furnished by this objective function, offers a means of exploration which is mathematically principled [1] and, efficient [14]. This addresses the exploration-exploitation dilemma without the need to add an ad-hoc information gain term [6]. And, this presents a particular advantage in sparse reward environments as compared to random exploratory methods [14]. The benefits of the active inference agent's exploratory objective, as compared to other exploratory motives such as maximum entropy RL [20], Boltzmann exploration [44] and ϵ-greedy methods [8], is left to our future review and, in particular, to

[58] One such (hybrid) architecture is presented by the Dyna framework [8, 12] (section 3.2.1). A non-RL hybrid (of iterative and amortised inference) in the context of variational autoencoders is described by the work of [46].

empirical study.

A second advantage of the active inference agent is the generality and flexibility afforded by the modelling framework. Specifically, perception, learning and action are unified via the free energy objective function. This accommodates various types of data and hierarchies of distributions where it is simple to extend active inference agents to POMDP models [6]. The modelling framework also offers the ability to represent preferences instead of rewards. While in this work, we have represented the prior preference distribution as one that prefers greater amounts of reward, it is also possible to remove this assumption by replacing it with a diffuse prior or, alternatively, preferences for other observations. Active inference agents, therefore, offer a Bayesian means of handling cases where rewards are not simply given and, are task-dependent, highly uncertain or, non-stationary [1]. The flexibility of the active inference framework, in this regard, offers an area of future research.

In summary, while this paper studies a means through which active inference and RL agents can be reconciled and, through which active inference agents can be scaled, it leaves many questions unanswered. Focusing on the machine learning framing of active inference agents and neglecting much of their neuroscience origins leaves a sizeable gap in our review of the framework. However, this framing has afforded an ease of understanding as to the mechanism of the active inference agent's inference schemes. After a review of tabular and deep RL methods, this framing has offered an accessible means at successfully approaching the problem of scaling active inference agents using deep RL methods.

Acknowledgements

I wish to express gratitude for the guidance, encouragement and tutelage of my two supervisors, Melusi and Dr. Shock. Both have been an inspiration to my studies, not only during this short work but, through undergraduate too. Melusi has exposed me to challenging work, always with an optimal mixture of enthusiasm and rigour. And, Dr. shock has grown a community of students and academics in which discussion and learning organically arise. While I have not met them in person for some while, I look forward to seeing them once again. Additionally, I wish to acknowledge the excellent work presented by Dr. Beren Millidge in his PhD titled *Applications of the Free Energy Principle to Machine Learning and Neuroscience* [1]. This work guided my study, answered many questions and, has left much inspiration for future work.

Declaration of Authorship

1. This work was done wholly while in candidature for a BSc Hons statistical sciences at the University of Cape Town.
2. The content of this work has not been previously submitted for a degree or any other qualification at this university or any other institution.
3. Where I have consulted the published work of others, this is always clearly attributed.
4. Where I have quoted from the work of others, the source is always given.
5. I have acknowledged all main sources of help.
6. I declare that this work is my own.

Signed:
R. Dubb

References

[1] B. Millidge, "Applications of the Free Energy Principle to Machine Learning and Neuroscience," 2021. [Online]. Available: https://arxiv.org/abs/2107.00140v1

[2] E. Schrodinger, *What is life?: the physical aspect of the living cell*. Trinity College Dublin, Ireland, 1944. [Online]. Available: http://www.whatislife.ie/schrodinger.htm

[3] K. Friston, J. Daunizeau, and S. J. Kiebel, "Reinforcement learning or active inference?" *PloS one*, vol. 4, no. 7, p. e6421, July 2009. [Online]. Available: https://europepmc.org/articles/PMC2713351

[4] K. Friston, J. Kilner, and L. Harrison, "A free energy principle for the brain," *Journal of Physiology-Paris*, vol. 100, no. 1, pp. 70–87, 2006, theoretical and Computational Neuroscience: Understanding Brain Functions. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S092842570600060X

[5] R. P. Feynman, *Statistical Mechanics A Set Of Lectures*. The Benjamin Cummings Publishing Company Inc., Reading, Massachusetts, 1998.

[6] L. D. Costa, N. Sajid, T. Parr, K. Friston, and R. Smith, "The relationship between dynamic programming and active inference: the discrete, finite-horizon case," 2020. [Online]. Available: https://arxiv.org/abs/2009.08111

[7] O. Berger-Tal, J. Nathan, E. Meron, and D. Saltz, "The Exploration-Exploitation Dilemma: A Multidisciplinary Framework," *PLOS ONE*, vol. 9, no. 4, pp. 1–8, 04 2014. [Online]. Available: https://doi.org/10.1371/journal.pone.0095693

[8] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. MIT Press, Cambridge, Massachusetts, 2018.

[9] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with Deep Reinforcement Learning," 2013. [Online]. Available: https://arxiv.org/abs/1312.5602

[10] R. S. Sutton, D. A. McAllester, S. Singh, and Y. Mansour, "Policy Gradient Methods for Reinforcement Learning with Function Approximation," in *NIPS*, 1999. [Online]. Available: https://papers.nips.cc/paper/1999/file/464d828b85b0bed98e80ade0a5c43b0f-Paper.pdf

[11] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous Methods for Deep Reinforcement Learning," 2016. [Online]. Available: https://arxiv.org/abs/1602.01783

[12] R. S. Sutton, "Dyna, an Integrated Architecture for Learning, Planning, and Reacting," *SIGART Bull.*, vol. 2, no. 4, p. 160–163, Jul. 1991. [Online]. Available: https://doi.org/10.1145/122344.122377

[13] B. Millidge, "Deep Active Inference as Variational Policy Gradients," 2019. [Online]. Available: https://arxiv.org/abs/1907.03876

[14] A. Tschantz, B. Millidge, A. K. Seth, and C. L. Buckley, "Reinforcement Learning through Active Inference," 2020. [Online]. Available: https://arxiv.org/abs/2002.12636

[15] R. Smith, K. Friston, and C. Whyte, "A Step-by-Step Tutorial on Active Inference and its Application to Empirical Data," *PsyArXiv*, 2021. [Online]. Available: https://doi.org/10.31234/osf.io/b4jm6

[16] L. Da Costa, T. Parr, N. Sajid, S. Veselic, V. Neacsu, and K. Friston, "Active inference on discrete state-spaces: A synthesis," *Journal of Mathematical Psychology*, vol. 99, p. 102447, Dec 2020. [Online]. Available: http://dx.doi.org/10.1016/j.jmp.2020.102447

[17] A. Tschantz, B. Millidge, A. K. Seth, and C. L. Buckley, "Control as Hybrid Inference," 2020. [Online]. Available: https://arxiv.org/abs/2007.05838

[18] M. Cullen, B. Davey, K. J. Friston, and R. J. Moran, "Active Inference in OpenAI Gym: A Paradigm for Computational Investigations Into Psychiatric Illness," *Biological Psychiatry: Cognitive Neuroscience and Neuroimaging*, vol. 3, no. 9, pp. 809–818, 2018, computational Methods and Modeling in Psychiatry. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2451902218301617

[19] M. Solms and K. Friston, "How and Why Consciousness Arises: Some Considerations From Physics and Physiology," *Journal of Consciousness Studies*, vol. 25, no. 5-6, pp. 202–238, 2018. [Online]. Available: https://discovery.ucl.ac.uk/id/eprint/10057681/

[20] S. Levine, "Reinforcement Learning and Control as Probabilistic Inference: Tutorial and Review," 2018. [Online]. Available: https://arxiv.org/abs/1805.00909

[21] B. Millidge, A. Tschantz, A. K. Seth, and C. L. Buckley, "Reinforcement learning as iterative and amortised inference," 2020. [Online]. Available: https://arxiv.org/abs/2006.10524v1#

[22] T. Parr and K. J. Friston, "The Anatomy of Inference: Generative Models and Brain Structure," *Frontiers in Computational Neuroscience*, vol. 12, p. 90, 2018. [Online]. Available: https://www.frontiersin.org/article/10.3389/fncom.2018.00090

[23] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "OpenAI Gym," 2016. [Online]. Available: https://arxiv.org/abs/1606.01540

[24] K. Friston, "Life as we know it," *Journal of the Royal Society, Interface / the Royal Society*, vol. 10, p. 20130475, 06 2013. [Online]. Available: https://doi.org/10.1098/rsif.2013.0475

[25] K. Friston and P. Ao, "Free energy, value, and attractors," *Computational and mathematical methods in medicine*, vol. 2012, p. 937860, 2012. [Online]. Available: https://europepmc.org/articles/PMC3249597

[26] R. Kaplan and K. J. Friston, "Planning and navigation as active inference," *Biological cybernetics*, vol. 112, no. 4, p. 323—343, August 2018. [Online]. Available: https://europepmc.org/articles/PMC6060791

[27] M. B. Mirza, R. A. Adams, C. D. Mathys, and K. J. Friston, "Scene Construction, Visual Foraging, and Active Inference," *Frontiers in computational neuroscience*, vol. 10, p. 56, 2016. [Online]. Available: https://europepmc.org/articles/PMC4906014

[28] K. Friston, "A free energy principle for a particular physics," 2019. [Online]. Available: https://export.arxiv.org/abs/1906.10184

[29] J. Pearl, "Chapter 3 - Markov and Bayesian Networks: Two Graphical Representations of Probabilistic Knowledge," in *Probabilistic Reasoning in Intelligent Systems*, J. Pearl, Ed. San Francisco (CA): Morgan Kaufmann, 1988, pp. 77–141. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780080514895500096

[30] T. Bayes, "An essay towards solving a problem in the doctrine of chances," *Phil. Trans. of the Royal Soc. of London*, vol. 53, pp. 370–418, 1763.

[31] S. Levine, "Deep Reinforcement Learning Lecture 4: Introduction to Reinforcement Learning, CS285 at UC Berkeley," University Lecture, 2020. [Online]. Available: http://rail.eecs.berkeley.edu/deeprlcourse/static/slides/lec-4.pdf

[32] K. Friston, F. Rigoli, D. Ognibene, C. Mathys, T. Fitzgerald, and G. Pezzulo, "Active inference and epistemic value," *Cognitive Neuroscience*, vol. 6, no. 4, pp. 187–214, 2015, pMID: 25689102. [Online]. Available: https://doi.org/10.1080/17588928.2015.1020053

[33] Oleg Solopchuk. Tutorial on Active Inference. [Online]. Available: https://medium.com/@solopchuk/tutorial-on-active-inference-30edcf50f5dc

[34] R. Bogacz, "A tutorial on the free-energy framework for modelling perception and learning," *Journal of mathematical psychology*, vol. 76, no. Pt B, p. 198—211, February 2017. [Online]. Available: https://europepmc.org/articles/PMC5341759

[35] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, pp. 484–, Jan. 2016. [Online]. Available: http://dx.doi.org/10.1038/nature16961

[36] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Aggressive driving with model predictive path integral control," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 1433–1440.

[37] M. A. Nielsen, *Neural Networks and Deep Learning*. Determination Press, 2015. [Online]. Available: http://neuralnetworksanddeeplearning.com.html

[38] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0893608089900208

[39] D. Silver, "Introduction to Reinforcement Learning with David Silver Lecture 7: Policy Gradient Methods," University Lecture, 2015. [Online]. Available: https://www.davidsilver.uk/wp-content/uploads/2020/03/pg.pdf

[40] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529—533, February 2015. [Online]. Available: http://europepmc.org/article/MED/25719670

[41] V. Feinberg, A. Wan, I. Stoica, M. I. Jordan, J. E. Gonzalez, and S. Levine, "Model-Based Value Estimation for Efficient Model-Free Reinforcement Learning," 2018. [Online]. Available: https://arxiv.org/abs/1803.00101

[42] J. Achiam, "Spinning Up in Deep Reinforcement Learning," 2018. [Online]. Available: https://spinningup.openai.com/en/latest/index.html

[43] S. Levine, "Deep Reinforcement Learning Lecture 10: Optimal Control and Planning, CS285 at UC Berkeley," University Lecture, 2020. [Online]. Available: http://rail.eecs.berkeley.edu/deeprlcourse/static/slides/lec-10.pdf

[44] N. Cesa-Bianchi, C. Gentile, G. Lugosi, and G. Neu, "Boltzmann Exploration Done Right," 2017. [Online]. Available: https://arxiv.org/abs/1705.10257

[45] K. Chua, R. Calandra, R. McAllister, and S. Levine, "Deep Reinforcement Learning in a Handful of Trials using Probabilistic Dynamics Models," 2018. [Online]. Available: https://arxiv.org/abs/1805.12114

[46] J. Marino, Y. Yue, and S. Mandt, "Iterative Amortized Inference," 2018. [Online]. Available: https://arxiv.org/abs/1807.09356

[47] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," 2018. [Online]. Available: https://arxiv.org/abs/1801.01290

[48] M. Okada, N. Kosaka, and T. Taniguchi, "PlaNet of the Bayesians: Reconsidering and Improving Deep Planning Network by Incorporating Bayesian Inference," 2020. [Online]. Available: https://arxiv.org/abs/2003.00370