Structures for Sophisticated Behaviour: Feudal Hierarchies and World Models

Sanjeevan Ahilan

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of

University College London.

Gatsby Computational Neuroscience Unit

University College London

December 22, 2020

I, Sanjeevan Ahilan, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

This thesis explores structured, reward-based behaviour in artificial agents and in animals. In Part I we investigate how reinforcement learning agents can learn to cooperate. Drawing inspiration from the hierarchical organisation of human societies, we propose the framework of Feudal Multi-agent Hierarchies (FMH), in which coordination of many agents is facilitated by a manager agent. We outline the structure of FMH and demonstrate its potential for decentralised learning and control. We show that, given an adequate set of subgoals from which to choose, FMH performs, and particularly scales, substantially better than cooperative approaches that use shared rewards.

We next investigate training FMH in simulation to solve a complex information gathering task. Our approach introduces a 'Centralised Policy Actor-Critic' (CPAC) and an alteration to the conventional multi-agent policy gradient, which allows one multi-agent system to advise the training of another. We further exploit this idea for communicating agents with shared rewards and demonstrate its efficacy.

In Part II we examine how animals discover and exploit underlying statistical structure in their environments, even when such structure is difficult to learn and use. By analysing behavioural data from an extended experiment with rats, we show that such hidden structure can indeed be learned, but also that subjects suffer from imperfections in their ability to infer their current state. We account for their behaviour using a Hidden Markov Model, in which recent observations are integrated imperfectly with evidence from the past. We find that over the course of training, subjects learn to track their progress through the task more accurately, a change that our model largely attributes to the more reliable integration of past evidence.

Impact Statement

The field of reinforcement learning seeks to develop and understand systems which can solve tasks by learning from experience. Recent work has demonstrated the potential for such systems to perform effectively in complex video game environments (Mnih et al., 2013; Vinyals et al., 2019) and on the ancient Chinese game of Go (Silver et al., 2016), where super-human performance was achieved. Applications in a variety of economically important domains is being actively explored, including autonomous driving (Sallab et al., 2017), resource management (Mao et al., 2016), finance (Moody and Saffell, 2001) and robotics (Kober et al., 2013).

In Part I of this thesis we propose a general method for learning in systems involving many reinforcement learning agents. Whilst our work is applied in abstract domains, it has influenced subsequent academic work which has applied this for traffic signal control (Ma and Wu, 2020) and fleet-management for ride sharing platforms (Jin et al., 2019) in simulation. In the future, similar methods may also be applied to other important problems, such as optimising cellular networks, smart grids and the allocation of computational resources. If such systems could be developed for real world use, they could greatly increase efficiency, reducing cost and providing better service to users, with considerable economic and societal advantages.

In Part II of this thesis we shed light on the decision making of animals by uncovering their use of a world model which reflects the nature of the task they are engaged in. Researchers in this field are actively investigating the flexibility with which such models can be learned in different situations (Behrens et al., 2018), and our findings provide a useful insight into this learning. Our results take the form of a computational model, which can account for animal behaviour in an interpretable way. Our results are useful not only to those seeking to model decision-making[1], but also for neuroscientists who seek to relate our results to the underlying activity of neural networks, and artificial intelligence researchers hoping to better understand the complex systems which they have developed.

[1] for example in humans, with associated economic consequences

Acknowledgements

Two roads diverged in a yellow wood,
And sorry I could not travel both
And be one traveler, long I stood
And looked down one as far as I could
To where it bent in the undergrowth

— Robert Frost, The Road Not Taken

First and foremost I would like to thank Peter Dayan. I was but a lowly serf, feudally[2] learning at his command. Over the course of my PhD I have unwisely diverged from his general supervisory direction many times, but despite this Peter has never stopped providing me with prompt, useful and occasionally brilliant feedback. I have enjoyed discussing and exploring so many topics with him over the years and am truly grateful for his patience throughout.

To my examiners, thank you for taking the time out of your no doubt busy schedules to read this thesis. I hope it is of interest to you, and I look forward to hearing your thoughts in the viva.

I am also thankful for my exceptional collaborators in the Shizgal lab at Concordia University. Peter Shizgal and the rest of the lab were very kind and welcoming when I visited, and without Rebecca Solomon's fine experimentation Part II of this thesis would not exist.

The Gatsby Unit has been a home to me these past few years and I am very lucky to have have been exposed to such a range of interesting people and ideas, and made some wonderful friends along the way. In particular I would like to mention

[2] or should I say 'futilely'?

Wenliang Li, Wenkai Xu, Heishiro Kanagawa, Alex Antrobus, Elena Zamfir, Ross Harper, Ricardo Monti, Jorge A. Menendez, Gopal Kotecha, Jesse Geerts, Virginia Rutten and Mehdi Keramati. I would also like to thank Wittawat Jitkrittum, without whom this thesis would have been completed two years earlier.

I am also glad to have made friends with others in the RL community, including those who I met on my internship as well as talented former masters students, including Rylan Schaeffer, Julius Kunze, Louis Kirsch and Danijar Hafner. Conferences also provided an opportunity to meet incredible people, and I would particularly like to thank Tian and Fernando from Rich Sutton's lab, who made my experience of RLDM 2017 so phenomenal. If I am ever asked if I travelled down the right road by starting a PhD, I will simply explain that I was once called 'Ratman' by Rich Sutton himself – and that has made all the difference.

I am also extremely grateful for my friends outside academia, in particular Deming Qin, Chris Gossage, Charmaine Law, Atchuthan Gopalan, Emily Olson, Ivan Seifert, Sonali Nundoochan, Neel Purmah, Krishna 'Nanthan' Jayanthiraa, Nimalan Kirubakaran and Bijoy Saha. They have kept my spirits high throughout and I hope to catch up with them more often in the future.

My wider family has also provided much support and affection, including my cousins, my uncles, my aunts, and my grandad Kanthappoo Paramothayan. Sadly, not all are still alive today and so I would like to dedicate this thesis to my grandmother, Ivy Kamala Pathmasundari Paramothayan, and my cousin Jegatheesan Rajarajan, whose life was tragically cut short by pancreatic cancer. I am thankful that he was able to show me his own PhD thesis before he passed – I know it meant a lot to him.

Finally, I would like to thank my immediate family, including my brother Arjunan, my sister-in-law Ahalya, Ammah and Appah. They have done so much for me not only during my PhD but throughout my life, and are a constant source of inspiration. I cannot express my appreciation in words; suffice to say I am eternally grateful.

Contents

List of Figures

List of Tables

Prologue

Chaos is merely order waiting to be deciphered

— José Saramago, The Double

Throughout human and animal history, organisms emerging into the world have frequently faced complexity and danger to bewildering degrees. Despite this, many go on to survive and even thrive, as they come to better understand, predict and control their environment.

How is such learning possible? One answer is that environments, whilst apparently chaotic on first encounter, are in fact highly structured in ways which can be exploited. This is true not only because natural laws are inherently simple, but also because humans and other animals have a tendency to seek order, imposing it on their lives in the face of uncertainty. Structures therefore are not only analytic, being inferred from direct experience, but also synthetic, being constructed to support future learning.

One source of structure is present in an animal's immediate environment. For example, streams always flow downhill according to gravity, predators are usually waiting at the nearby waterhole and the birds will begin singing just as the sun rises. The ability to flexibly learn and use models of how the world behaves is one of the most interesting aspects of animal intelligence. Models allow animals to infer relevant states of the world and make predictions about the future over a range of timescales. The model itself need not be veridical but merely useful for the animal deciding on what to do next.

Once a decision has been made, the resulting behaviours themselves may utilise recurring motifs, in deference to the recurring structures present in the envi-

ronment. This has been of substantial interest to reinforcement learning researchers, studying how such flexible problem solving capabilities may be imbued into artificial systems by composing behavioural units flexibly to solve larger problems.

The problems themselves take place on a stage which is ever changing, as agents interact and learn from their interactions. No agent is an island, and despite the complexity of social interaction, most can benefit from collaboration – if organised the right way. Structure can therefore be used to foster cooperation, making interactions more predictable and coordinated.

In this thesis we explore two different topics which share the unifying theme of structure and its influence on learning. The structures involved are disparate in both cases and we in general consider these topics separately, dividing the thesis into two parts before reviewing the sum total of our findings.

In Part I we consider artificial learning agents interacting in a multi-agent system, and organise their interactions by invoking a managerial hierarchy. Our approach, which we call 'Feudal Multi-agent Hierarchies' draws inspiration from the field of hierarchical reinforcement learning, which seeks to learn effective decompositions of problems and behaviours. We train our system on problems which require cooperation and coordination, and show that our method can scale to many agents. We also explore situations in which the information for decision making is not immediately available and must be communicated, introducing a 'Centralised Policy Actor-Critic' which resolves this difficulty.

In Part II, we investigate the behaviour of rats seeking reward in a self-stimulation task with a hidden transition structure. We examine the resulting 'world model' they learn and illuminate cases where their inferences are imperfect due to partial observability. By modelling their model of the task, we show how, with experience, subjects can adaptively learn to integrate past evidence to achieve more accurate predictions.

Overall, we hope this tale of two topics provides insight into their respective fields as well as the varied influences and roles structure can play in natural and artificial systems. The intersection of animal behaviour and artificial intelligence is

a fascinating one, and we hope our findings will be illuminating to readers from a variety of perspectives.

Chapter 1

Background

In this chapter we introduce and review the fundamentals of single-agent reinforcement learning (RL)[1]. We then describe model-free RL and introduce 'deep' RL models which use artificial neural networks as function approximators. Finally, we introduce latent variable models and the problem of partial observability.

1.1 Fundamentals

1.1.1 The RL paradigm

The beginning of an animal's life is often characterised by weakness, a lack of skill and, in many cases, reliance on parents to provide sustenance. Given time and repeated interactions with their environment, however, animals will generally acquire the necessary skills for survival, whether this be hunting prey or navigating complex terrain. The field of reinforcement learning (RL) (Sutton and Barto, 2018) concerns itself with the computational principles underlying this kind of goal-directed learning through interaction. Rather than directly theorising about how people or animals learn, it explores idealised learning situations and evaluates the performance of various learning methods.

Reinforcement learning has a rich history spanning multiple fields. In psychology it can be used to model classical (Pavlovian) and operant (instrumental) conditioning. In neuroscience it has been used to model the dopamine system of the

[1] our review draws from David Silver's UCL course available at `http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html` and Sutton and Barto's textbook (Sutton and Barto, 2018)

brain. Modelling decision-making is also highly relevant to economics, in particular fields such as bounded rationality, and in engineering it has extensive overlap with the field of optimal control. In mathematics, investigation has continued under the guise of operations research and of course much research into RL is explored by those in computer science. The plethora of perspectives ensures that RL continues to be an exciting and extraordinarily interdisciplinary field.

1.1.2 Agent and environment

RL problems typically draw a separation between the agent and the environment. The agent receives observation o_t and scalar reward r_t from the environment and emits action a_t, where t indicates the time step. The environment receives action a_t from the agent and then emits a reward r_{t+1} and an observation o_{t+1}. The cycle then begins again with the agent emitting its next action.

How the environment responds to the agent's action is determined by the environment state s_t, which is updated at every time step. The conditional distribution for the next environment state depends only on the present state and action and therefore satisfies the Markov property:

$$P(s_{t+1}|s_t,a_t) = P(s_{t+1}|s_1,\ldots,s_t,a_1,\ldots,a_t) \tag{1.1}$$

The environment state is in general private from the agent, which only receives observations and rewards. The conditional distribution for the next observation given the current observation is not in general Markov, and so it may be beneficial for an agent to construct its own notion of state s_t^α, which it uses to determine its next action. This can be defined as $s_t^\alpha = f(h_t)$, where h_t is the history of the agent's sequence of observations, actions and rewards:

$$h_t = a_1, o_1, r_1, \ldots, a_t, o_t, r_t \tag{1.2}$$

1.1.3 Observability

A special case exists when the observation received by the agent o_t is identical to the environment state s_t (such that there is no need to distinguish between the two). This is the assumption underlying the formalism of Markov decision processes covered in the next section. An environment is partially observable if the agent cannot observe the full environment state, meaning that the conditional distribution for its next observation given its current observation does not satisfy the Markov property. This assumption underlies the formalism of a partially observable Markov decision process which we describe in Section 1.4.3.

1.1.4 Markov processes and Markov reward processes

A Markov process (or Markov chain) is a sequence of random states with the Markov property. It is defined in terms of the tuple $\langle \mathcal{S}, \mathcal{P} \rangle$ where $\mathcal{S}$ is a finite set of states and $\mathcal{P} : \mathcal{S} \times \mathcal{S} \to [0,1]$ is the state transition probability kernel.

A Markov Reward Process (MRP) $\langle \mathcal{S}, \mathcal{P}, r, \gamma \rangle$ extends the Markov process by including a reward function $r : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ for each state transition and a discount factor γ. The immediate expected reward in a given state is defined as: $r(s) = \sum_{s'} \mathcal{P}(s,s') r(s,s')$.

The discount factor $\gamma \in [0,1]$ is used to determine the present value of future rewards. Conventionally, a reward received k steps into the future is of worth γ^k times what it would be worth if received immediately. As we will shortly see, the cumulative sum of discounted rewards is a quantity RL agents often seek to maximise, and so $\gamma < 1$ ensures that this sum is bounded (assuming r is bounded).

1.1.5 Markov decision processes

Single-agent RL can be formalised in terms of Markov decision processes (MDPs). The idea of an MDP is to capture the key components available to the learning agent; the agent's sensation of the state of its environment, the actions it takes which can affect the state, and the rewards associated with states and actions. An MDP extends the formalism of an MRP to include a finite set of actions on which both $\mathcal{P}$ and r depend. Discrete-time, infinite-horizon MDPs are described in terms of the 5-tuple

$\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma \rangle$ where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$ is the state transition probability kernel, $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the immediate reward function and $\gamma \in [0,1)$ is the discount factor. The expected immediate reward for a given state and action is defined as $r(s,a) = \sum_{s'} \mathcal{P}(s,a,s') r(s,a,s')$, which we use for convenience subsequently.

1.1.6 Policies, values and models

Common components of a reinforcement learning agent are a policy, value function and a model. The policy $\pi : \mathcal{S} \times \mathcal{A} \to [0,1]$ is the agent's behaviour function which denotes the probability of taking action a in state s. Agents may also act according to a deterministic policy $\mu : \mathcal{S} \to \mathcal{A}$. We will assume that policies are stochastic unless otherwise noted.

Given an MDP and a policy π, the observed state sequence is a Markov process $\langle \mathcal{S}, \mathcal{P}^{\pi} \rangle$. Similarly, the state and reward sequence is a MRP $\langle \mathcal{S}, \mathcal{P}^{\pi}, r_{\pi}, \gamma \rangle$ in which:

$$\mathcal{P}^{\pi}(s,s') = \sum_{a \in \mathcal{A}} \pi(s,a) \mathcal{P}(s,a,s') \tag{1.3}$$

$$r_{\pi}(s) = \sum_{a \in \mathcal{A}} \pi(s,a) r(s,a) \tag{1.4}$$

Starting from any particular state s at time step $t = 0$, the value function $v_{\pi}(s)$ is a prediction of the expected discounted future reward given that the agent starts in state s and follows policy π:

$$v_{\pi}(s) = \mathbb{E}_{\pi} \left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} | s_0 = s \right] \tag{1.5}$$

where $r_{t+1} = r(s_t, a_t, s_{t+1})$

which is the solution of an associated Bellman expectation equation:

$$v_{\pi}(s) = \sum_{a \in A} \pi(s,a) \left[r(s,a) + \gamma \sum_{s' \in S} \mathcal{P}(s,a,s') v_{\pi}(s') \right] \tag{1.6}$$

In matrix form the Bellman expectation equation can be expressed in terms of the induced MRP:

$$\boldsymbol{v}_\pi = \boldsymbol{r}_\pi + \gamma \mathcal{P}^\pi \boldsymbol{v}_\pi = (\mathcal{I} - \gamma \mathcal{P}^\pi)^{-1} \boldsymbol{r}_\pi \tag{1.7}$$

where $\boldsymbol{v}_\pi \in \mathbb{R}^{|\mathcal{S}|}$ and $\boldsymbol{r}_\pi \in \mathbb{R}^{|\mathcal{S}|}$ are the vector of values and expected immediate rewards respectively for each state under policy π. We can also define a Bellman expectation backup operator:

$$T^\pi(\boldsymbol{v}) = \boldsymbol{r}_\pi + \gamma \mathcal{P}^\pi \boldsymbol{v} \tag{1.8}$$

which has a fixed point of $\boldsymbol{v}^\pi$.

An action-value for a policy π can also be defined, which is the expected discounted future reward for executing action a and subsequently following policy π.

$$\begin{aligned} q_\pi(s,a) &= r(s,a) + \gamma \sum_{s' \in S} \mathcal{P}(s,a,s') v_\pi(s') \\ &= r(s,a) + \gamma \sum_{s' \in S} \mathcal{P}(s,a,s') \sum_{a' \in \mathcal{A}} \pi(s',a') q_\pi(s',a') \end{aligned} \tag{1.9}$$

The process of estimating v_π or q_π is known as policy evaluation. Policies can be evaluated without directly knowing or estimating a model, using instead the directly sampled experience of the environment, an approach which is known as 'model-free'. However a 'model-based' approach is also possible in which a model is used to predict what the environment will do next. A key component of a model is an estimate of $\mathcal{P}(s,a,s')$, the probability of the next state given the current state and action. Another is an estimate of $r(s,a)$, the expected immediate reward.

Policy evaluation enables a value function to be learned for a given policy. However, in reality we wish to learn the best possible policy. The value function for this is known as the optimal value function and corresponds to the maximum value function over all policies:

$$v_*(s) = \max_\pi v_\pi(s) \tag{1.10}$$

The definition of the optimal action-value function (which evaluates the immediate action a in state s) is similarly:

$$q_*(s,a) = \max_{\pi} q_\pi(s,a) \tag{1.11}$$

A partial ordering over policies can be defined according to:

$$\pi \geq \pi' \text{ if } v_\pi(s) \geq v_{\pi'}(s), \forall s \tag{1.12}$$

For any MDP there exists an optimal policy π_* that is better than or equal to all other policies. All optimal policies achieve the optimal value function and optimal action-value function and there is always a deterministic optimal policy for any MDP. The latter is achieved by selecting:

$$a = \arg\max_{a \in \mathcal{A}} q_*(s,a) \tag{1.13}$$

If there are many possible actions which satisfy this, any of these may be chosen to constitute an optimal policy (of which there may be many). The optimal value and state-value functions satisfy Bellman optimality equations:

$$\begin{aligned} v_*(s) &= \max_{a \in \mathcal{A}} q_*(s,a) \\ v_*(s) &= \max_{a \in \mathcal{A}} \Big[r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}(s,a,s') v_*(s') \Big] \\ q_*(s,a) &= r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}(s,a,s') \max_{a'} q_*(s',a') \end{aligned} \tag{1.14}$$

The Bellman optimality equation is non-linear with no closed form solution (in general). Solving it therefore requires iterative solution methods.

1.1.7 Dynamic programming

Dynamic programming (DP) (Bertsekas et al., 1995) refers to a collection of algorithms that can be used to compute optimal policies given a perfect model of the environment as an MDP. In general, DP solves complex problems by breaking them down into subproblems and then combining the solutions. It is particularly useful for overlapping subproblems, the solutions to which reoccur many times when solving the overall problem, making it more computationally efficient to cache and

reuse them.

When applied to MDPs, the recursive decomposition of DP corresponds to the Bellman equation and the cached solution to the value function. DP assumes that the MDP is fully known and therefore does not address the full RL problem but instead addresses the problem of planning. By planning, the prediction problem can be addressed by finding the value function v_π of a given policy π. This can be evaluated by iterative application of the Bellman Expectation Backup (Equation 1.8).

This leads to convergence to a unique fixed point v_π, which can be shown using the contraction mapping theorem (also known as the Banach fixed-point theorem) (Banach, 1922). When a Bellman expectation backup operator T^π is applied to two value functions $\boldsymbol{u}$ and $\boldsymbol{v}$ over states, we find that it is a γ-contraction:

$$\begin{aligned} ||T^\pi(\boldsymbol{u}) - T^\pi(\boldsymbol{v})||_\infty &= ||(r_\pi + \gamma \mathcal{P}^\pi \boldsymbol{u}) - (r_\pi + \gamma \mathcal{P}^\pi \boldsymbol{v})||_\infty \\ &= ||\gamma \mathcal{P}^\pi (\boldsymbol{u} - \boldsymbol{v})||_\infty \\ &\leq ||\gamma \mathcal{P}^\pi \mathbf{1} ||\boldsymbol{u} - \boldsymbol{v}||_\infty||_\infty \\ &\leq \gamma ||\boldsymbol{u} - \boldsymbol{v}||_\infty \end{aligned} \tag{1.15}$$

where $\mathbf{1}$ is a vector of ones and the infinity norm of a vector $\boldsymbol{a}$ is denoted $||\boldsymbol{a}||_\infty$ and is defined as the maximum value of its components. This contraction ensures that both $\boldsymbol{u}$ and $\boldsymbol{v}$ converge to the unique fixed point of T^π which is $\boldsymbol{v}_\pi$.

For control, DP can be used to find the optimal value function v_* and in turn the optimal policy π_*. One possibility is *policy iteration* in which the current policy π is first evaluated as described and then subsequently improved to π' such that:

$$\pi'(s) = \arg\max_{a \in \mathcal{A}} q_\pi(s,a) \tag{1.16}$$

This improves the value from any state s over one step:

$$q_\pi(s, \pi'(s)) = \max_{a \in \mathcal{A}} q_\pi(s,a) \geq \sum_{a \in \mathcal{A}} \pi(s,a) q_\pi(s,a) = v_\pi(s) \tag{1.17}$$

It can be shown that this improves the value function such that that $v_{\pi'}(s) \geq$

$v_\pi(s)$ (Silver, 2015). This process is then repeated, with improvements ending when the Bellman optimality equation (1.14) has been satisfied and convergence to π_* achieved. A generalisation of policy iteration is also possible in which, instead of waiting for policy evaluation to converge, only n steps of evaluation are taken before policy improvement occurs and the process is repeated. If $n = 1$ this is known as *value iteration*, as the policy is no longer explicit (being a direct consequence of the value function). Like policy iteration, value iteration is also guaranteed to converge to the optimal value function and policy. This can be demonstrated using the contraction mapping theorem.

1.2 Model-free approaches

1.2.1 Prediction

As has been outlined, dynamic programming can be used to solve known MDPs enabling optimal value functions and policies to be found. However, in many cases the MDP is not directly known - instead an agent taking actions in the MDP must learn directly from its experiences, as it transitions from state to state and receives rewards accordingly. One approach, known as 'model-free', seeks to solve MDPs without learning transitions or rewards. For prediction, a key quantity to estimate in this setting is the expected discounted future reward. A sampled estimate of this, starting from state s_t, is known as the return:

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + ... = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \tag{1.18}$$

which depends on the actions sampled from the policy, and states from transitions.

Monte-Carlo (MC) methods seek to estimate this directly using complete episodes of experience. Introducing a learning rate α_t, the agent's value function can therefore be updated according to[2]:

$$v(s_t) \leftarrow v(s_t) + \alpha_t \left[R_t - v(s_t)\right] \tag{1.19}$$

[2]assuming a table-based representation rather than use of a function approximator

The value function updated in this way will converge to a solution with minimum mean-square error (best fit to the observed returns), assuming a suitable sequential decrease in the learning rate.

Temporal-difference (TD) learning methods learn from incomplete episodes by bootstrapping. For example, if learning occurs after a single step, this is known as TD(0), which has the following update:

$$v(s_t) \leftarrow v(s_t) + \alpha_t \left[r_{t+1} + \gamma v(s_{t+1}) - v(s_t) \right] \tag{1.20}$$

where $r_{t+1} + \gamma v(s_{t+1})$ is known as the *target*. This approximates the full-width Bellman expectation backup (Equation 1.8) in which every successor state and action is considered, with experiences instead being sampled. TD(0) will converge to the solution of the maximum likelihood Markov model which best fits the data (again assuming a suitable sequential decrease in the learning rate). This solution may be different from the minimum mean-square error solution of MC methods, which do not assume the Markov property.

Unlike MC methods, TD methods introduce bias into the estimated return as the currently estimated value function may be different from the true value function. However, they generally have reduced variance relative to MC methods, as in MC the estimated return depends on a potentially long sequence of random actions, transitions and rewards.

The distinction between MC and TD methods can be blurred by considering multi-step TD methods (rather than only TD(0)), in which rewards are sampled for a number of steps before the value function is used to compute an estimate of future rewards. The n-step return is defined as:

$$R_t^{(n)} = r_{t+1} + \gamma r_{t+2} + \ldots + \gamma^{n-1} r_{t+n} + \gamma^n v(s_{t+n}) \tag{1.21}$$

As $n \to \infty$ it tends towards the unbiased MC return. An algorithm may seek to find a good bias-variance tradeoff by estimating a weighted combination of n-step returns; one popular method to do this is known as TD(λ):

$$R_t^{\lambda} = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} R_t^{(n)} \tag{1.22}$$

where $\lambda \in [0,1]$.

1.2.2 Control with action-value functions

Model free control concerns itself with optimising rather than evaluating the RL objective. Policies may be evaluated according to various objectives. In the case of continuing environments, the objective can be the average value or the average reward per time-step. We however focus on episodic environments, assuming an initial distribution over starting states $p_0(s) : \mathcal{S} \rightarrow [0,1]$. The objective is thus:

$$J(\pi) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} | p_0(s)\right] \tag{1.23}$$

Note that if the domain of the starting state distribution is only over a single starting state, the objective is simply the value function (Equation 1.5) in that starting state. This objective can equivalently be expressed as:

$$J(\pi) = \mathbb{E}_{s\sim\rho^{\pi}, a\sim\pi}[r(s,a)] \tag{1.24}$$

where:

$$\rho^{\pi}(s) := \sum_{s'}\sum_{t=0}^{\infty} \gamma^t p(s_t = s | s', \pi) p_0(s') \tag{1.25}$$

is the improper discounted state distribution induced by policy π starting from an initial state distribution $p_0(s')$. We will later describe policy gradient methods which seek to optimise this objective directly.

However, we first consider model-free approaches which rely on an action-value function $q(s,a)$ to achieve control (a value function $v(s)$ alone is insufficient for model-free control). The optimal action-value function $q_*(s,a)$ must be learned, with MC and TD methods both viable. Once it has been learned, an optimal policy may be achieved by selecting the best action in each state (Equation 1.13).

However, unlike dynamic programming, full-width backups are not used and

so if actions are selected greedily (meaning those with highest action-values are always chosen) then certain states and actions may never be correctly evaluated. Model-free RL methods must therefore allow for enough exploration during learning before ultimately exploiting this learning to achieve near-optimal cumulative reward.

One simple approach, known as ε-greedy is to take a random action with probability ε but otherwise act greedily according to the current estimate of the action-value function. The value of ε can be decreased with the number of episodes. This can satisfy a condition known as greedy in the limit of infinite exploration in which all state-action pairs are explored infinitely many times and the policy converges to the greedy policy.

One popular algorithm for model-free control is known as Q-learning, which seeks to learn the optimal action-value function whilst using a policy which also takes exploratory actions (such as epsilon greedy). This learning is termed *off-policy* as the policy used to sample experience is different from the policy being learned (the optimal policy). The resulting update is:

$$q(s_t,a_t) \leftarrow q(s_t,a_t) + \alpha\left[r_{t+1} + \gamma \max_{a' \in \mathcal{A}} q(s_{t+1},a') - q(s_t,a_t)\right] \tag{1.26}$$

An alternative to off-policy Q-learning is on-policy SARSA. This uses the sampled sampled state s_t, action a_t, reward r_{t+1}, next state s_{t+1}, and next action a_{t+1} for updates[3]:

$$q(s_t,a_t) \leftarrow q(s_t,a_t) + \alpha(r_{t+1} + \gamma q(s_{t+1},a_{t+1}) - q(s_t,a_t)) \tag{1.27}$$

1.2.3 Value function approximation

So far we have assumed a tabular representation of states and actions such that each state is separately updated. However, in practice we would like value functions and policies to generalise to new states and actions, and so it is beneficial to use

[3] and also gives SARSA its name

function approximators such as deep neural networks. A common approach is to approximate the value function or action-value function:

$$\begin{aligned} v_w(s) &= \hat{v}(s;w) \approx v_\pi(s) \\ q_w(s,a) &= \hat{q}(s,a;w) \approx q_\pi(s,a) \end{aligned} \tag{1.28}$$

where w are the parameters we wish to learn. If we start by assuming we know the true value function v_π, we can define a mean square error between the approximate value function and the true function:

$$\mathcal{L}(w) = \mathbb{E}_\pi[(v_\pi(s) - v_w(s))^2] \tag{1.29}$$

Given a distribution of states $s \sim p(s)$[4], we can minimise this iteratively using stochastic gradient descent:

$$w \leftarrow w + \alpha(v_\pi(s_t) - v_w(s_t))\nabla_w v_w(s_t) \tag{1.30}$$

In reality we can only use a better estimate of v_π provided by the sampled reward(s). For example, if we use the TD(0) target the update is:

$$w \leftarrow w + \alpha(r_{t+1} + \gamma v_w(s_{t+1}) - v_w(s_t))\nabla_w v_w(s_t) \tag{1.31}$$

Updates like this are known as 'semi-gradient' as the gradient of the value function used to define the target is ignored.

If we use a linear function approximator $v_w(s) = x(s)^T w$ (where features $x(s)$ and w are vectors), then we find:

$$w \leftarrow w + \alpha(r_{t+1} + \gamma v_w(s_{t+1}) - v_w(s_t))x(s_t) \tag{1.32}$$

indicating that the linear weights are updated in proportion to the activity of their corresponding features. Non-linear function approximators can also be used, but typically have weaker convergence guarantees than linear function approxima-

[4] we later discuss a method for sampling states

tors. Nevertheless, due to their flexibility such approximators have enabled impressive performance in a number of challenging domains, such as Atari games and Go.

1.2.4 Policy gradient methods

Parameterised stochastic policies π_θ may be improved using the *policy gradient theorem* (Sutton et al., 2000). This can be derived for any of the common RL objectives. To demonstrate a derivation of this result we use a starting state objective $J(\theta) = v_{\pi_\theta}(s_0)$ with a single starting state s_0:

$$\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta v_\pi(s_0) \\
&= \nabla_\theta \sum_a \pi(s_0,a) q_\pi(s_0,a) \\
&= \sum_a \nabla_\theta \pi(s_0,a) q_\pi(s_0,a) + \pi(s_0,a) \nabla_\theta q_\pi(s_0,a) \\
&= \sum_a \nabla_\theta \pi(s_0,a) q_\pi(s_0,a) + \pi(s_0,a) \nabla_\theta \left[r(s_0,a) + \sum_{s'} \gamma \mathcal{P}(s_0,a,s') v_\pi(s') \right] \\
&= \sum_a \nabla_\theta \pi(s_0,a) q_\pi(s_0,a) + \pi(s_0,a) \sum_{s'} \gamma \mathcal{P}(s_0,a,s') \nabla_\theta v_\pi(s')
\end{aligned} \tag{1.33}$$

We note that we could continue to unroll $\nabla_\theta v_\pi(s')$ on the R.H.S in the same way as we have already done. Considering now transitions from starting state s_0 to arbitrary state s we therefore find:

$$\nabla_\theta v_\pi(s_0) = \sum_s \sum_{t=0}^{\infty} \gamma^t p(s_t = s | s_0, \pi) \sum_a \nabla_\theta \pi(s,a) q_\pi(s,a) \tag{1.34}$$

where $\sum_{t=0}^{\infty} \gamma^t p(s_t = s|s_0,\pi)$ is the discounted state distribution $\rho^\pi(s)$ from a fixed starting state s_0 (Equation 1.25). This derivation holds even when there is a distribution over starting states, and gives us the policy gradient theorem:

$$\nabla_\theta J(\theta) = \sum_s \rho^\pi(s) \sum_a \nabla_\theta \pi(s,a) q_\pi(s,a) \tag{1.35}$$

Using the likelihood ratio trick:

$$\begin{aligned}\nabla_\theta \pi(s,a) &= \pi(s,a)\frac{\nabla_\theta \pi(s,a)}{\pi(s,a)} \\ &= \pi(s,a)\nabla_\theta \log \pi(s,a)\end{aligned} \tag{1.36}$$

this can be equivalently expressed as:

$$\begin{aligned}\nabla_\theta J(\theta) &= \sum_s \rho^\pi(s) \sum_a \pi(s,a) q_\pi(s,a) \nabla_\theta \log \pi(s,a) \\ &= \mathbb{E}_\pi[q_\pi(s,a)\nabla_\theta \log \pi(s,a)]\end{aligned} \tag{1.37}$$

The policy gradient theorem result enables model-free learning as gradients need only be determined for the policy rather than for properties of the environment. There are a variety of approaches for determining q_π. If q_π is approximated using the sample return (Equation 1.18), this leads to the algorithm known as REINFORCE (Williams, 1992):

$$\theta \leftarrow \theta + \alpha R_t \nabla_\theta \log \pi(s_t, a_t) \tag{1.38}$$

As there is no bootstrapping here, this is also known as MC policy gradient. An alternative approach is to separately approximate q_π with a 'critic' q_w giving rise to what are commonly known as 'actor-critic' methods. These introduce two sets of parameter updates; the critic parameters w are updated to approximate q_π, and the policy (actor) parameters θ are updated according to the policy gradient as indicated by the critic. The critic itself can be updated according to the TD error. An example of this approach is SARSA actor-critic:

$$\begin{aligned}w &\leftarrow w + \alpha_1(r_{t+1} + \gamma q_w(s_{t+1}, a_{t+1}) - q_w(s_t, a_t))\nabla_w q_w(s_t, a_t) \\ \theta &\leftarrow \theta + \alpha_2 q_w(s_t, a_t)\nabla_\theta \log \pi(s_t, a_t)\end{aligned} \tag{1.39}$$

where different learning rates α_1 and α_2 may be used for the actor and the critic.

1.2.5 Baselines

Whether we use REINFORCE or an actor-critic based approach to policy gradients, it is possible to reduce the variance further by the introduction of baselines. If this baseline depends only on the state s, then we find it introduces no bias:

$$\begin{aligned}\sum_s \rho^\pi(s) \sum_a \nabla_\theta \pi(s,a) b(s) &= \sum_s \rho^\pi(s) b(s) \nabla_\theta \sum_a \pi(s,a) \\ &= \sum_s \rho^\pi(s) b(s) \nabla_\theta 1 \\ &= 0\end{aligned} \tag{1.40}$$

A natural choice for the state-dependent baseline is the value function:

$$\begin{aligned}\nabla_\theta J(\theta) &= \mathbb{E}_\pi[(q_\pi(s,a) - v_\pi(s)) \nabla_\theta \log \pi(s,a)] \\ &= \mathbb{E}_\pi[A_\pi(s,a) \nabla_\theta \log \pi(s,a)]\end{aligned} \tag{1.41}$$

where A_π is known as the advantage, which may in some algorithms be approximated directly (rather than approximating both q_π and v_π).

1.2.6 Compatible function approximation

In the general case, our choice to approximate q_π with q_w introduces bias such that there are no guarantees of convergence to a local optimum. However, in the special case of a compatible function approximator we can introduce no bias and take steps in the direction of the true policy gradient. This becomes possible when the critic's function approximator reaches a minimum in the mean-squared error:

$$\begin{aligned}0 &= \mathbb{E}_\pi[\nabla_w (q_\pi(s,a) - q_w(s,a))^2] \\ &= \mathbb{E}_\pi[(q_\pi(s,a) - q_w(s,a)) \nabla_w q_w(s,a)]\end{aligned} \tag{1.42}$$

If we choose $q_w(s,a)$ such that $\nabla_w q_w(s,a) = \nabla_\theta \log \pi(s,a)$ we find:

$$\mathbb{E}_\pi[q_\pi(s,a) \nabla_\theta \log \pi(s,a)] = \mathbb{E}_\pi[q_w(s,a) \nabla_\theta \log \pi(s,a)] \tag{1.43}$$

where the L.H.S is equal to the true policy gradient and so our function approximation has introduced no bias. For example, if the policy is a Boltzmann policy with a linear combination of features, then a compatible value function must be linear in the same features as the policy except normalised to zero mean for each state using a subtractive baseline (Sutton et al., 2000).

1.2.7 Deterministic policy gradients

Deterministic policy gradients (DPG) is a frequently used single-agent algorithm for continuous control using model-free RL (Silver et al., 2014). It is an actor-critic method which uses deterministic policies $\mu_\theta : \mathcal{S} \to \mathcal{A}$. The parameters of the policy, θ, are adjusted in an off-policy fashion using an exploratory behavioural policy to perform stochastic gradient ascent on an objective $J(\theta) = \mathbb{E}_{s\sim\rho^\mu, a\sim\mu_\theta}[r(s,a)]$, where ρ^μ is the discounted state distribution induced by policy μ starting from an initial state distribution with density p_0.

The DPG algorithm builds off the deterministic policy gradient theorem:

$$\nabla_\theta J(\theta) = \mathbb{E}_{s\sim\rho^\mu}[\nabla_\theta \mu_\theta(s) \nabla_a q_\mu(s,a)|_{a=\mu_\theta(s)}]. \tag{1.44}$$

approximating q_μ by the critic q_w, which is differentiable in the action and updated using Q-learning:

$$\delta_t = r_{t+1} + \gamma q_w(s_{t+1}, \mu_\theta(s_{t+1})) - q_w(s_t, a_t)$$
$$w \leftarrow w + \alpha_1 \delta_t \nabla_w q_w(s_t, a_t)$$

The parameters of the policy are then updated according to:

$$\theta \leftarrow \theta + \alpha_2 \nabla_\theta \mu_\theta(s_t) \nabla_a q_w(s_t, a_t)|_{a=\mu_\theta(s_t)} \tag{1.45}$$

One advantage of DPG over stochastic off-policy actor-critic methods is that the integral over actions is removed. This allows DPG to avoid importance sampling in the actor, which can result in better sample efficiency. To explore actions stochastically with its behavioural policy, DPG uses a noisy version of its determin-

istic policy, for example by adding Gaussian noise.

1.3 Deep reinforcement learning

The policies and value functions used in reinforcement learning can be learned using artificial neural network function approximators. When such networks have many layers they are conventionally denoted as 'deep', and are typically trained on large amounts of data using stochastic gradient descent (LeCun et al., 2015). The application of deep networks in model-free reinforcement learning garnered extensive attention when they were successfully used to learn a variety of Atari games from scratch (Mnih et al., 2013). For the particular problem of learning from pixels a convolutional neural network architecture was used (LeCun et al., 1998), which are highly effective at extracting useful features from images. They have been extensively used on supervised image classification tasks due to their ability to scale to large and complex datasets (LeCun et al., 2015).

The application of deep reinforcement learning (DRL) required the overcoming of major technical challenges. These include the high degree of correlation between states encountered by an RL agent and the non-stationarity of the data distribution as the agent learns. Two approaches to address these issues are 'experience replay' and 'target networks', which we describe in this section. We then explain a DRL algorithm we frequently use in this thesis, known as deep deterministic policy gradients (DDPG), and an approach which can be used to generate differentiable samples from a categorical distribution known as Gumbel Softmax, which we later use with DRL policies.

1.3.1 Experience replay

As an agent interacts with its environment it receives experiences that can be used for learning. However, rather than using those experiences immediately, it is possible to store such experience in a 'replay buffer' and sample them at a later point in time for learning. The benefits of such an approach were introduced by Mnih et al. (2013) for their 'deep Q-learning' algorithm. At each timestep, this method stores experiences $e_t = (s_t, a_t, r_{t+1}, s_{t+1})$ in a replay buffer over many episodes. Af-

ter sufficient experience has been collected, Q-learning updates are then applied to randomly sampled experiences from the buffer. This breaks the correlation between samples, reducing the variance of updates and the potential to overfit to recent experience.

1.3.2 Target networks

When using temporal difference learning with deep function approximators a common challenge is stability of learning. A source of instability arises when the same function approximator is used to evaluate both the value of the current state and the value of the target state for the temporal difference update. After such updates, the approximated value of both current and target state change (unlike tabular methods), which can lead to a runaway target. To address this, deep RL algorithms often make use of a separate target network that remains stable even whilst the standard network is updated. As it is not desirable for the target network to diverge too far from the standard network's improved predictions, at fixed intervals the parameters of the standard network can be copied to the target network. Alternatively, this transition is made more slowly using Polyak averaging:

$$\phi_{targ} \leftarrow \rho\phi_{targ} + (1-\rho)\phi \tag{1.46}$$

where ϕ are the parameters of the standard network and ρ is a hyperparameter typically close to 1.

1.3.3 Deep deterministic policy gradients

Deep deterministic policy gradients (DDPG) (Lillicrap et al., 2015) is a variant of DPG with policy μ_θ and critic q_w being represented by deep neural networks. DDPG built off the success off DQN (Mnih et al., 2015), which succeeded in stabilising training through use of a replay buffer and target networks.

In addition to this, their implementation of DDPG also utilised batch normalisation (Ioffe and Szegedy, 2015) (a method for re-centering and re-scaling input layers in deep networks) and exploration using noise generated by an Ornstein-Uhlenbeck process (although later implementations often found equal success with

Gaussian noise).

1.3.4 Re-parameterisation with Gumbel softmax

Deep networks are commonly used to parameterise probabilistic models, which can be used to generate samples. For example, a deep network can be used as a function approximator for an RL agent's stochastic policy whose samples correspond to actions. In this setting, it is sometimes the case that we wish to compute the gradient of a sample from the model with respect to model parameters. Re-parameterisation refers to approaches which achieve this by rewriting the probabilistic model as a deterministic function plus a random noise term (Glasserman and Ho, 1991; Kingma and Welling, 2013). Samples are generated by first sampling random noise and then computing the deterministic transformation specified by the model, enabling gradients to be backpropagated.

An example of a reparameterisation is the Gumbel Softmax estimator, which provides a general solution to obtaining gradients from samples of the categorical distribution (Jang et al., 2016; Maddison et al., 2016). The Gumbel Softmax builds on the Gumbel-Max trick (Gumbel, 1954) which, given a categorical distribution with class probabilites π, allows one to draw samples z from it by taking:

$$z = \text{one\_hot}(\arg\max_i [g_i + \log \pi_i]) \tag{1.47}$$

where g_i are samples drawn from Gumbel(0,1)[5].

For the Gumbel softmax, the one-hot argmax operation is replaced by a softmax approximation with a temperature parameter τ, to generate k-dimensional sample vectors $y \in \Delta^{n-1}$ on the simplex, where:

$$y_i = \frac{\exp((g_i + \log \pi_i)/\tau)}{\sum_{j=1}^{k} \exp((g_j + \log \pi_j)/\tau)} \tag{1.48}$$

for $i = 1, \ldots, k$.

When the temperature is zero, samples from the Gumbel softmax distribution become one-hot and the Gumbel softmax distribution becomes identical to the cat-

[5] using the procedure $u \sim \text{Uniform}(0,1)$ and computing $g = -\log(-\log(u))$

egorical distribution. For temperatures above zero, samples are no longer one-hot and have a well defined gradient $\frac{\partial y}{\partial \pi}$, enabling backpropagation to be used to compute gradients for weight updates.

The so-called Straight-Through Gumbel Estimator applies the Gumbel-Max trick (with argmax) on the forward pass to generate a discrete sample, but uses the continuous approximation to it, the Gumbel Softmax, on the backward pass, to compute a biased estimate of the gradients. This is useful if the problem only allows for discrete samples (even during training). The temperature can be optimised as a hyperparameter; the higher the temperature used the greater the bias, but the lower the variance of the estimated gradient.

1.4 Latent variables and partial observability

1.4.1 Latent variable models

Hidden or 'latent' variables correspond to variables which are not directly observed but nevertheless influence observed variables and thus may be inferred from observation. In reinforcement learning, it can be beneficial for agents to infer latent variables as these often provide a simpler and more parsimonious description of the world, enabling better predictions of future states and thus more effective control.

Latent variable models are common in the field of unsupervised learning. Given data $p(x)$ we may describe a probability distribution over x according to:

$$p(x;\theta_{x|z},\theta_z) = \int dz\; p(x|z;\theta_{x|z})p(z;\theta_z) \tag{1.49}$$

where $\theta_{x|z}$ parameterises the conditional distribution $x|z$ and θ_z parameterises the distribution over z.

Key aims in unsupervised learning include capturing high-dimensional correlations with fewer parameters (as in probabilistic principal components analysis), generating samples from a data distribution, describing an underlying generative process z which describes causes of x, and flexibly modelling complex distributions even when the underlying components are simple (e.g. belonging to an exponential family).

1.4.2 Hidden Markov models

Hidden Markov models (HMMs) are an example of a discrete latent variable model for time series. In an HMM, a probability distribution over sequences of observations $\boldsymbol{o} = \{o_1, \ldots, o_t, \ldots, o_T\}$ is defined by invoking a corresponding sequence of latent discrete state variables $\boldsymbol{s} = \{s_1, \ldots, s_t, \ldots, s_T\}$. The sequence of hidden states has Markov dynamics - i.e. given s_t, s_τ is independent of s_ρ for all $\rho < t < \tau$.

The HMM is defined by two sets of parameters, the transition matrix $\mathcal{P}(s'|s)$ and an emission distribution $\Omega(o|s)$, which defines the probability of an observation given the underlying state[6]. Given a sequence of random variables $o_{1:T}$, the joint probability factorises according to:

$$p(o_{1:T}, s_{1:T}) = p(s_1)\Omega(o_1|s_1)\prod_{t=2}^{T}\mathcal{P}(s_t|s_{t-1})\Omega(o_t|s_t) \tag{1.50}$$

where s_t corresponds to the discrete latent state at time t. This factorisation highlights the Markov structure of latent variables in the HMM. Crucially however, the observed variables need not be Markov, and may be either continuous or discrete.

A common method for estimating the parameters which define $\mathcal{P}$ and Ω involves first taking an expectation over the posterior probability of hidden state sequences. As the length of a time series increases the number of possible latent states $s_{1:T}$ grows exponentially, which makes naive approaches to inference costly. Fortunately, the Markov structure of latent variables in the HMM nevertheless allows efficient inference using the forward-backward algorithm, which is based on the principle of dynamic programming.

Once hidden states have been inferred using the current setting of the parameters, the parameters themselves can be optimised. The alternating process of inferring hidden states given observations and the current setting of the parameters, and then optimising parameters given the inferred posterior is known as the Expectation Maximisation (EM) algorithm (Dempster et al., 1977), which for the special case of HMMs is known as the Baum-Welch algorithm.

[6]These may additionally be conditioned on the action, in which case they are known as input-output HMMs.

1.4.3 Partially observable Markov decision processes

A partially observable Markov decision process (POMDP) (Kaelbling et al., 1998) is a generalisation of an MDP in which the agent cannot directly observe the true state of the system, the dynamics of which is determined by an MDP. Formally, a POMDP is a 7-tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, r, \mathcal{O}, \Omega, \gamma \rangle$ where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$ is the state transition probability kernel, $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the reward function, $\mathcal{O}$ is the set of observations, $\Omega : \mathcal{S} \times \mathcal{A} \times \mathcal{O} \to [0,1]$ is the observation probability kernel and $\gamma \in [0,1)$ is the discount factor. As with MDPs, agents in POMDPs seek to learn a policy $\pi(s_t^a)$ which maximises some notion of cumulative reward, commonly $\mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r_{t+1}]$. This policy depends on the agent's representation of state $s_t^a = f(h_t)$, which is a function of its history.

One approach to solving POMDPs is by maintaining a belief state over the underlying state of the world - as in an HMM, this underlying state is Markovian. Thus maintaining a belief over states only requires knowledge of the previous belief state, the action taken and the current observation. Beliefs may then be updated according to:

$$b'(s') = \eta \Omega(o'|s',a) \sum_{s \in \mathcal{S}} \mathcal{P}(s'|s,a) b(s) \tag{1.51}$$

where $\eta = 1/\sum_{s'} \Omega(o'|s',a) \sum_{s \in \mathcal{S}} \mathcal{P}(s'|s,a) b(s)$ is a normalising constant.

However, in practice, maintaining belief states in POMDPs will be computationally intractable for any reasonably sized problem. In order to address this, approximate solutions may be used. Alternatively, methods which use recurrent neural networks can enable agents to construct their own state representations, which may in turn enable relevant aspects of the state to be approximately Markov.

Part I

Feudal Hierarchies

If at first you don't succeed, try management

Chapter 2

Introduction

An important challenge in the field of artificial intelligence is understanding how interacting agents in a shared environment can collectively learn to optimise a task objective. Such problems can correspond to a wide range of relevant situations, such as agents in a warehouse collaborating to manufacture as many goods as possible, or driverless cars negotiating a busy intersection such that no individual car is left waiting too long. Ideally we would like to develop systems which can solve multi-agent tasks such as these simply by learning from experience, rather than requiring precise specification of behaviours, which is likely impossible for most non-trivial tasks.

An appropriate formalism is that of multi-agent reinforcement learning (MARL) (Busoniu et al., 2008), which operates at the intersection of reinforcement learning (RL) and multi-agent systems research. This treats each agent in the system as a reinforcement learning agent, responsible for its own actions separate from other agents acting in the environment. This can be contrasted with problems which involve only one agent, or those which involve a centralised controller which chooses the actions of all agents. Learning in the MARL setting introduces its own set of difficulties, as well as advantages, which has spurred much research in the field.

One of the critical considerations in MARL is the non-stationarity of the environment from the perspective of any one agent. Unlike the case where there is only one RL agent, this occurs even when the physical or simulated environment is

fixed, because the learning of other agents causes them to change the actions they take. This affects the distribution of state transitions and rewards perceived by each agent, which cannot ordinarily observe the actions of other agents.

In addition to partial observability of actions, MARL problems often involve partial observability of state. Each agent may have different observations which inform them about different aspects of the environment but which are individually insufficient for success on a particular task. One way of resolving the difficulties of distributed state can involve agents communicating with each other, but the appropriate strategy to achieve this must be learned.

We also would like our systems to learn even when the number of agents is large – a problem of scalability. MARL's assumption that actions are controlled by separated learning agents rather than a single centralised controller is potentially advantageous in this situation, as the joint action space of all agents grows exponentially. However, when addressing a single task objective, increasing the number of agents makes credit-assignment difficult, as the overall reward becomes an increasingly noisy reflection of a given agent's contributions (Wolpert and Tumer, 2002; Chang et al., 2004).

In Part I we seek to address these challenges by introducing a new framework for multi-agent learning termed Feudal Multi-Agent Hierarchies (FMH). Our idea is inspired by the structure of real-life managerial hierarchies, in which different tasks are assigned to workers, which agree to toil towards them for the benefit of the collective. In such a setting, multiplicity is often seen as a blessing rather than a curse – the potential to achieve more collectively than would otherwise be possible. However, in order for chaos not to simply ensue, the manager must learn to assign tasks and coordinate its workers to optimise overall task performance.

In the following chapters, we describe FMH in detail and demonstrate its application. Our framework concerns itself mainly with alterations of reward and state, and can in principle be combined with a variety of multi-agent algorithms. To test our idea, we utilise the DDPG algorithm (Section 1.3.3) and the MADDPG algorithm (which we later describe), and conduct experiments in the multi-agent particle

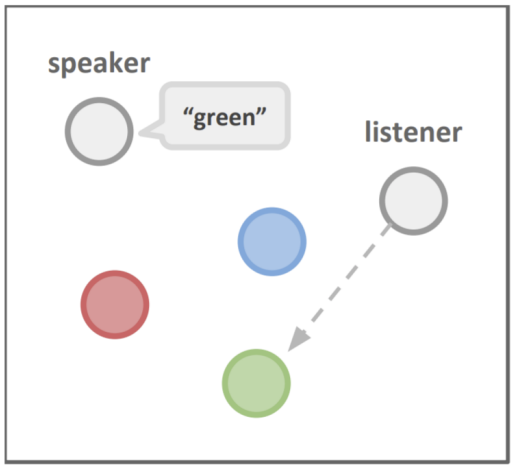

Figure 2.1: Cooperative Communication. From Figure 2, Lowe et al. (2017). The immobile Speaker sees the colour of the target landmark (green in this case). It communicates a discrete message to the Listener at every time step. Both agents receive reward proportional to the negative distance of the Listener from the target.

environment (MPE)[1]. The MPE is a simple world with a continuous observation and discrete action space, along with some basic simulated physics. It allows for many agents to interact simultaneously in a shared environment but with different perspectives. To navigate, each agent must learn a non-trivial control policy, and to aid cooperation, each agent may send and receive discrete costless communication messages.

We focus in particular on problems involving partial observability, such as those where an agent cannot observe the information it needs to solve a task but where this information is available to another agent which can communicate with it. An example of such a problem, introduced by Lowe et al. (2017), is known as Cooperative Communication (Figure 2.1), in which there are two cooperative agents, one called the 'Speaker' and the other called the 'Listener', placed in an environment with three landmarks of differing colours. On each episode, the Listener must navigate to a randomly selected landmark; and both agents obtain reward proportional to its negative distance from this target. However, whilst the Listener observes its relative position from each of the differently coloured landmarks, it does not know which landmark is the target. Instead, the colour of the target landmark can be

[1] https://github.com/openai/multiagent-particle-envs

seen by the Speaker, which cannot observe the Listener and is unable to move. The Speaker can however communicate to the Listener at every time step, and so successful performance on the task corresponds to it helping the Listener to reach the correct target. An interesting aspect of this problem is that the communication policy of the Speaker is likely to change over time, as it learns to communicate, and this non-stationarity can make learning difficult for the Listener, which must learn to act based on the messages it receives.

An approach which can prove effective for the Cooperative Communication problem was introduced by Lowe et al. (2017) which involved a method known as 'centralised training'. This is a method for MARL training in simulation which uses the observations and actions of other agents to help each agent learn. It has become an important paradigm for training MARL systems in simulation, and we describe it in detail later in this chapter.

Our method of FMH can be applied in both a decentralised and centralised context. In Chapter 3 we explore the idea of FMH in a decentralised setting whereas in Chapter 4 we investigate the centralised case. The structure of Part I is therefore a follows:

- In Chapter 3 we describe our new framework of Feudal Multi-agent Hierarchies and demonstrate how it can be used in the decentralised setting. We show how our method can be used to address the challenges of non-stationarity and coordination, enabling it to scale to large numbers of agents.

- In Chapter 4 we investigate applications of centralised training. We develop a new method which involves a centralised policy actor-critic (CPAC) and a modification to the conventional multi-agent policy gradient. We apply our method to information-gathering tasks and demonstrate the potential of our general strategy when applied in both the fully cooperative setting and for FMH.

- In Chapter 5 we describe how our work relates to other work in the field, and its limitations. We also discuss future research directions.

The remainder of the current chapter is a review of relevant research in this field. We divide this into two sections; multi-agent RL, which is our domain of application, and hierarchical RL, a field which provided inspiration for our proposed approach.

2.1 Multi-agent RL

In recent years, MARL has generated substantial interest due to successful applications of single-agent reinforcement learning, such as DeepMind's deep Q-Learning algorithm which successfully learned to play Atari games from scratch (Mnih et al., 2013) and subsequent work on the long-standing challenge of the competitive board game Go, for which a super-human standard was achieved (Silver et al., 2016). More recently, DeepMind incorporated ideas from multi-agent learning to train a population of agents which achieved super-human performance on the real-time strategy game Starcraft (Vinyals et al., 2019)[2]. This approach ensured that each agent was robust to a diversity of opponent strategies, a hallmark of elite play. MARL has also been applied effectively in the mixed cooperative-competitive domain, with large-scale experiments involving collaborating teams against competitors outperforming humans on various videogames (Jaderberg et al., 2019) and eliciting advanced behaviours such as tool use (Baker et al., 2019).

Despite these successes, MARL systems can be very difficult to train, due to difficulties explained in the introduction. Here we review important concepts in MARL relevant to our research, including the complexities of multi-agent interaction and approaches towards a solution. We focus in particular on cooperative reinforcement learning (Panait and Luke, 2005), in which agents must collectively work towards a single task objective.

2.1.1 Definitions

2.1.1.1 Partially Observable Markov Games

An appropriate formalism for MARL systems in which agents cannot directly observe the full environmental state is known as a partially observable Markov

[2] constraints on AI reaction time were chosen to ensure an approximately level playing field

game (POMG) (Littman, 1994; Hu et al., 1998)[3]. This extends the notion of a POMDP (Section 1.4.3) to the case where there are multiple agents. A POMG for N agents is defined by a set of states $\mathcal{S}$, and sets of actions $\mathcal{A}_1,\ldots,\mathcal{A}_N$ and observations $\mathcal{O}_1,\ldots,\mathcal{O}_N$ for each agent. In general, the stochastic policy of agent i may depend on the set of action-observation histories $H_i \equiv (\mathcal{O}_i \times \mathcal{A}_i)^*$ such that $\pi_i : \mathcal{H}_i \times \mathcal{A}_i \to [0,1]$. In this work we restrict ourselves to history-independent stochastic policies $\pi_i : \mathcal{O}_i \times \mathcal{A}_i \to [0,1]$ as these are sufficient to solve the tasks we examine. We denote the set of possible policies for agent i as $\Delta(\mathcal{O}_i, \mathcal{A}_i)$. We assume agents act synchronously in an environment with discrete time and that the next state is generated according to $\mathcal{P} : \mathcal{S} \times \mathcal{A}_1 \times \ldots \times \mathcal{A}_n \times \mathcal{S} \to [0,1]$, which is the state transition probability kernel.

Each agent i obtains deterministic rewards defined as $r_i : \mathcal{S} \times \mathcal{A}_1 \times \ldots \times \mathcal{A}_n \times \mathcal{S} \to \mathbb{R}$ and (for our experiments) receives a deterministic private observation $o_i : \mathcal{S} \to \mathcal{O}_i$. There is an initial state distribution $\rho_0 : \mathcal{S} \to [0,1]$ and each agent aims to maximise its own discounted sum of future rewards. We denote the joint strategy of all the agents as $\boldsymbol{\pi}$ and the joint strategy of all agents except agent i as $\boldsymbol{\pi}_{-i}$. We use the notation $\langle \pi_i, \boldsymbol{\pi}_{-i} \rangle$ to refer to the joint strategy where agent i follows π_i while the other agents follow their policy from $\boldsymbol{\pi}_{-i}$.

A key aspect of multi-agent RL is whether it is cooperative or competitive. This is determined by reward: in a fully cooperative POMG all agents share the same reward, whereas in a fully competitive two-player POMG the agents have opposing rewards. All other games are known as mixed.

2.1.1.2 Nash equilibrium and Pareto-optimality

Multi-agent interactions have benefited from a wealth of research from the field of game theory. We therefore briefly introduce key game theoretic concepts, assuming for convenience a fully-observable Markov Game in which agents observe the true state s. The value function for agent i is:

$$v_{\pi_i}(s) = \mathbb{E}_{\boldsymbol{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t r_{i,t+1} | s_0 = s\right] \tag{2.1}$$

[3] also known as a stochastic game

where $r_{i,t+1}$ is player i's reward at time $t+1$.

A joint policy $\boldsymbol{\pi}^*$ defines a *Nash equilibrium* in a Markov game iff, for each agent i:

$$\forall \pi_i \in \Delta(\mathcal{S}, \mathcal{A}_i), \forall s \in \mathcal{S}, \quad v_{\langle \pi_i^*, \boldsymbol{\pi}^*_{-i} \rangle}(s) \geq v_{\langle \pi_i, \boldsymbol{\pi}^*_{-i} \rangle}(s) \tag{2.2}$$

such that no agent can improve its discounted cumulative reward by unilaterally deviating from a Nash equilibrium. A Markov game may have many Nash equilibria, however, there may exist 'better' solutions which are not Nash yet are of greater value for each agent than any of the Nash solutions. It is therefore useful to define a separate notion of *Pareto optimality*. A joint policy $\hat{\boldsymbol{\pi}}$ Pareto-dominates another joint policy $\boldsymbol{\pi}$, written $\hat{\boldsymbol{\pi}} > \boldsymbol{\pi}$, iff in all states:

$$\forall i, \forall s \in \mathcal{S}, \quad v_{i,\hat{\boldsymbol{\pi}}}(s) \geq v_{i,\boldsymbol{\pi}}(s) \quad \textit{and} \quad \exists j, \exists s \in \mathcal{S}, \quad v_{j,\hat{\boldsymbol{\pi}}}(s) > v_{j,\boldsymbol{\pi}}(s) \tag{2.3}$$

If a joint policy $\hat{\boldsymbol{\pi}}^*$ is not Pareto-dominated by any other joint strategy, then $\hat{\boldsymbol{\pi}}^*$ is Pareto-optimal. This means that no agent's expected gain can be improved upon without decreasing the expected gain of any other agent. For games which are not fully cooperative the Pareto-optimal solution will not necessarily correspond to a Nash equilibrium and vice-versa (such as in the Prisoner's dilemma). However, in fully-cooperative games the Pareto-optimal solution will of course correspond to a Nash equilibrium (as shared rewards means unilateral deviation must be no better for the deviating agent).

2.1.2 Interaction concepts

2.1.2.1 Coordination

The challenge of coordinating multiple agents can be illustrated by considering a fully cooperative Markov Game. Whilst we typically treat the actions of other agents as unobservable, consider the case where all actions can in fact be observed and there are no private observations such that only a single corresponding action-value function $Q(s, a_1, \ldots, a_N)$ need be learned. For multi-agent control, actions are

selected separately by each agent and this results in a coordination problem even when accurate action-values have been learned. This is because the simple strategy of selecting an optimal action with the assumption that other agents also take optimal actions relies on there only being a single optimal action for each agent. In cases where there are multiple actions of equal value available to other agents, a given agent cannot know the other actions which will be taken and so cannot coordinate to the optimal joint action. From a game-theoretic perpsective, this is known as the Pareto-selection problem and arises from the existence of multiple Pareto-optimal Nash equilibria.

Another difficulty of coordination is caused by the exploration of many learning agents. This can interfere with MARL systems trying to solve tasks which require precise coordination, as the chance of all agents exploiting at a given moment diminishes as the number of agents increases (Matignon et al., 2012). In such situations it may be more likely for agents to converge towards a sub-optimal Nash equilibrium less reliant on precise coordination, even whilst there exist jointly deterministic policies which would be better.

2.1.2.2 Communication

One approach which can alleviate issues of coordination is allowing agents to communicate with each other. One way to classify communication is whether it is explicit or implicit. In the explicit case, communication can be modeled as being separate from the environment. For example, Lowe et al. (2019) divide the action space of agent i into disjoint environment actions $\mathcal{A}_i^e$ and communication actions $\mathcal{A}_i^m$, such that $\mathcal{A}_i^e \cup \mathcal{A}_i^m = \mathcal{A}_i$ and $\mathcal{A}_i^e \cap \mathcal{A}_i^m = \emptyset$. Environment actions directly affect the environment dynamics and the rewards of the agents whereas communication actions do not, but are instead only observed by other agents. Communication may be targeted to specific agents or broadcast to all agents and may be costly or incur zero cost. The zero cost formulation is commonly used and referred to as 'cheap talk', a term which originated from the game theory community.

Implicit communication corresponds to multi-agent problems which do not have separate communication channels but nevertheless allow for communicative

behaviours to other agents through changes of their environment. An example of this kind of communication would be the leaving of 'breadcrumbs' in the environment by one agent, for another agent to follow, or teaching by imitation.

In this work, we focus on explicit communication with the expectation that dedicated communication channels will be frequently integrated into artificial multi-agent systems. Interestingly, communication has the potential to blur the boundaries between cooperative MARL and single-agent RL as, in the unrestricted case, it allows such systems simply to be treated as single-agent systems (Panait and Luke, 2005). To understand this, consider a multi-agent system of $N + 1$ agents, where the extra agent is denoted special status. If the N regular agents instantaneously communicate their observations to the special agent, which views them all and immediately communicates back to each of the N agents the desired action it wishes that agent to take, then the problem is equivalent to training with a single centralised agent. This removes the complexities of coordination and non-stationarity common in MARL and enables straightforward application of single-agent methods[4].

Of course, in most MARL problems there are restrictions on communication, even when a communicative action is not assumed to be costly in terms of reward. Common limitations include bottlenecks, such as only allowing the communication of a few discrete symbols, and unreliable messages due to noise or lag. The most effective use of communication amongst multi-agent teams in such situations can be difficult to directly prescribe but the hope is that multi-agent RL could learn to effectively use such available channels. Indeed, recent work does suggest that this is possible, with communication enabling better performance than when such channels are removed (Lowe et al., 2019).

Recent work in multi-agent RL has leveraged differentiable communication channels to improve performance on fully cooperative problems further. A differentiable channel allows for gradients to be backpropagated such that agents are able to assign credit for their communication to the downstream behaviour of other agents (Sukhbaatar et al., 2016; Foerster et al., 2016; Peng et al., 2017; Mordatch

[4] in practice the exponential scaling of the action space with the number of agents may make such an approach ineffective

and Abbeel, 2018). If the 'real' communication channel is in fact discrete, the differentiable channel may be discretized (Foerster et al., 2016). However, this discretisation can induce differences in training and test performance, and more effective learning has been observed when using instead the Straight-Through Gumbel-softmax estimator (Havrylov and Titov, 2017).

Whilst learning communication via backpropagation is an interesting strategy, we do not rely on the existence of a differentiable communication channel between agents in this work. We instead learn communication using each agent's individual policy gradient, as in other recent work (Lowe et al., 2017; Iqbal and Sha, 2019).

2.1.3 Solution concepts

2.1.3.1 Decentralised model-free learning

A straightforward way to address multi-agent problems is to use single agent model-free RL algorithms for each agent, such as Q-Learning (Watkins and Dayan, 1992) or policy gradients (Williams, 1992; Sutton et al., 2000). In this 'independent learners' approach each agent receives only local observations (as defined by the environment) which are used as input to the agent's policy and to train its parameters (Tan, 1993; Gupta et al., 2017). A potential benefit of such an approach are the minimal assumptions used compared to more complex methods, and the simplicity of use. It can therefore be applied in a variety of scenarios, both in simulation and in the real world. The downside, however, is that the non-stationarity of multi-agent RL removes convergence guarantees present in the single-agent case (assuming the internal states of other agents cannot be observed).

2.1.3.2 Centralised learning with decentralised control

Independent learners only have access to local action and observation histories during training. However, modern approaches to training and evaluating performance often take place at least partially in simulation, which allows precise control over the information available to the agents. Consequently, methods have been developed which exploit this potential by using the actions and observations of all agents for training. Some approaches go further still by providing agents with access to the

environmental state, rendering a partially-observable problem as fully observable.

Although it is restrictive for agents to require access to this full information, this approach has generated recent interest due to the potential for centralised training of ultimately decentralised policies (Lowe et al., 2017; Foerster et al., 2018). For example, in simulation one can train a critic for each agent which exploits the action and observation histories of all agents to aid the training of local policies for each one. These policies can then be deployed in multi-agent systems in the real world, where decentralised execution is required.

Recent advances have demonstrated the potential for strong performance of centralised methods when training in simulation is viable. In general multi-agent learning causes the environment to be non-stationary from the perspective of each agent. However, centralised learning allows for the critic(s) to depend on the environmental state and joint action of all agents and thus learn to account for this non-stationarity. To see why this is the case, we consider the policy gradient theorem (Equation 1.37) in the multi-agent setting when a single critic is used (as in Foerster et al. (2018)). We denote the vector of joint actions for all agents as $\boldsymbol{a}$, of all agent policy parameters as θ, and individual agent observations and actions as o_i and a_i respectively. Assuming the centralised critic has access to the state s which renders transitions Markov, the multi-agent policy gradient can be expressed as:

$$\begin{aligned}\nabla_\theta J(\theta) &= \mathbb{E}_{\boldsymbol{\pi}}\Big[\sum_i q_{\boldsymbol{\pi}}(s,\boldsymbol{a})\nabla_\theta \log \pi(o_i,a_i)\Big] \\ &= \mathbb{E}_{\boldsymbol{\pi}}\Big[q_{\boldsymbol{\pi}}(s,\boldsymbol{a})\nabla_\theta \log \prod_i \pi(o_i,a_i)\Big]\end{aligned} \tag{2.4}$$

Writing the joint policy as a product of independent actors:

$$\boldsymbol{\pi}(s,\boldsymbol{a}) = \prod_i \pi(o_i,a_i) \tag{2.5}$$

we therefore recover the standard single-agent actor-critic policy gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\boldsymbol{\pi}}\Big[q_{\boldsymbol{\pi}}(s,\boldsymbol{a})\nabla_\theta \log \boldsymbol{\pi}(s,\boldsymbol{a})\Big] \tag{2.6}$$

Following Konda and Tsitsiklis (2000) this will converge to a local maximum

of $J^{\boldsymbol{\pi}}$ under strict conditions; the policy must be differentiable, the update timescales for $q_{\boldsymbol{\pi}}$ and $\boldsymbol{\pi}$ must be sufficiently slow, with $\boldsymbol{\pi}$ being updated sufficiently slower than $q_{\boldsymbol{\pi}}$, and $q_{\boldsymbol{\pi}}$ must use a compatible representation with $\boldsymbol{\pi}$[5].

The convergence proof used relies on the critic being centralised. However, this may not be possible if learning outside of simulation or if the multi-agent system requires further training after deployment due to changes in the environment. In addition, such methods may struggle to scale as the number of agents increases, due to the curse of dimensionality.

2.1.3.3 MADDPG

Multi-agent deep deterministic policy gradients (MADDPG) (Lowe et al., 2017) is an algorithm for centralised training and decentralised control of multi-agent systems. It uses deterministic polices, as in DDPG (Section 1.3.3), which condition only on each agent's local observations and actions.

MADDPG handles the non-stationarity associated with the simultaneous adaptation of all the agents by introducing a separate centralised critic $Q_i^{\boldsymbol{\mu}}(\boldsymbol{o}, \boldsymbol{a})$ for each agent where $\boldsymbol{\mu}$ corresponds to the set of deterministic policies $\mu_i : \mathcal{O} \rightarrow \mathcal{A}$ of all agents. Here we have denoted the vector of joint observations for all agents as $\boldsymbol{o}$. In addition inputs to the centralised critic could be further augmented by aspects of the environmental state not observed by any of the agents, although this was not explored in the original wrok.

The multi-agent policy gradient for policy parameters θ of agent i is:

$$\nabla_{\theta_i} J(\theta_i) = \mathbb{E}_{\boldsymbol{o},\boldsymbol{a}\sim\mathcal{D}}[\nabla_{\theta_i}\mu_i(o_i)\nabla_{a_i} Q_i^{\boldsymbol{\mu}}(\boldsymbol{o}, \boldsymbol{a})|_{a_i=\mu_i(o_i)}]. \tag{2.7}$$

where $\mathcal{D}$ is the experience replay buffer which contains the tuples $(\boldsymbol{o}, \boldsymbol{a}, \boldsymbol{r}, \boldsymbol{o'})$ and $\boldsymbol{r} = (r_1, \ldots, r_N)$. Like DDPG, each $Q_i^{\boldsymbol{\mu}}$ is approximated by a critic Q_i^w which is updated to minimise the error with the target.

$$\mathcal{L}(w_i) = \mathbb{E}_{\boldsymbol{o},\boldsymbol{a},\boldsymbol{o'},\boldsymbol{a'}\sim\mathcal{D}}[(Q_i^w(\boldsymbol{o}, \boldsymbol{a}) - y)^2] \tag{2.8}$$

[5] of course, amongst practitioners such conditions are rarely observed

where $y = r_i + \gamma Q_i^w(\boldsymbol{o}', \boldsymbol{a}')$ is evaluated for the next state and action (as stored in the replay buffer, and in practice using the critic target network).

MADDPG allows for each agent to have differing rewards and therefore Lowe et al., trained agents on a variety of mixed cooperative-competitive environments in the MPE, significantly improving upon agents trained independently with DDPG. They utilised history-independent feedforward networks for critics and policies, and the output of the latter was transformed into a sample from the categorical distribution using the Gumbel Softmax (Section 1.3.4), as discrete actions are used for control in the MPE.

2.2 Hierarchical RL

The field of hierarchical RL (HRL) (Barto and Mahadevan, 2003) originally arose to address the difficulties faced by single-agent RL. In our work, we attempt to marry these ideas with the field of MARL, but first we outline the relevant single-agent work. The introduction of HRL was first conceived to address the curse of dimensionality which, even with the use of parameteric function approximators, still looms large in a variety of problems. Furthermore, it seeks to address the problems of sparse reward, in which obtaining even a single reward on a novel task may require complex sequences of actions, exacerbating the problems of exploration and temporal credit-assignment.

HRL attempts to address these challenges by noting that the world (and indeed agents themselves) are highly structured such that the temporally-extended behaviours needed to obtain reward in one problem often share substantial similarities with required behaviours in other problems. Thus if 'good' behavioural sub-policies can be learned for a given task, a higher-level policy need only learn to combine these on subsequent tasks rather than learning again from scratch.[6] This temporal abstraction would therefore lead to more effective exploration, and credit-assignment, and so more efficient learning. There are a plethora of approaches to HRL, for which we only outline a few. Our choice is determined by their relation-

[6]Of course, similar reasoning may be used to motivate sub-sub-policies and so on.

ship to our own method as well as their historical signficance.

2.2.1 Feudal RL

One of the earliest approaches to HRL is Feudal RL (FRL) (Dayan and Hinton, 1993; Watkins, 1989; Dayan, 1994)[7], which establishes a managerial hierarchy of agents 'inside the head' of a single agent. In this approach, high-level managers learn to set goals to their sub-managers who, in turn, learn how to satisfy them by setting their own goals to sub-sub-managers and so on. Only the lowest level (currently-active) agent, which we call the 'worker', takes regular (primitive) actions in the world, with the actions of higher levels in the hierarchy corresponding to goals for the agent one level-down. Agents are rewarded according to the extent to which they satisfy the goals they receive, except for the highest-level manager which receives no explicit goal but instead receives the reward defined by the task.

The motivation for introducing this substantial multi-agent complexity to a single agent problem is the hope that explicit managerial goal-setting will allow lower level agents to learn useful component behaviours. In turn the manager can learn to sequentially select goals (and thus behaviours) to solve a variety of tasks.

In the original implementation, FRL decomposed a 2D gridworld into squares of increasing size, such that higher-level managers view a state-space which is smaller but has coarser resolution. The goals available to managers are to request the transition of sub-managers to adjacent states (from its coarser perspective) or to search for reward in the existing state. If this goal is satisfied, the manager provides reward to its sub-manager, otherwise the goal eventally reaches a timeout and no reward is provided.

FRL utilises the principles of 'information-hiding' and 'reward-hiding'. Information-hiding corresponds to the fact that higher level managers do not know the actions that workers take to achieve the goals which are set and furthermore have a coarser view of the state. This latter approach is used to address the curse of dimensionality but does so at the expense of making the task non-Markovian

[7]The name refers to the 'feudal' system, common in medieval kingdoms such as England and Japan, as well as various academic institutions.

at higher levels, which can render problems insoluble.[8] Nevertheless, it raises the interesting idea that different levels of a hierarchy might have different state representations. The second principle of reward-hiding is that sub-managers only report to managers one level above them, such that any other rewards, including task rewards, are irrelevant to them. The sub-manager therefore focuses only on achieving the smaller-scale task it has been set rather than having to contend with the larger problem (for which learning might be very slow). The experiments used for FRL showed that whilst initial learning on problems was slower than ordinary Q-learning, learning on subsequent problems was greatly sped up due to the effective reuse of subpolicies.

2.2.2 Options

This framework addresses the problem of HRL by introducing the notion of temporally-extended actions, known as options (Sutton et al., 1999). The initial work sought to find a minimal extension to the RL framework which would allow for a general treatment of temporally abstract knowledge and action. It achieved this by building on the theory of semi-Markov decision processes (SMDPs) (Bradtke and Duff, 1995) and considering their interplay with MDPs. SMDPs are a special kind of MDP in which actions are temporally-extended and variable in duration, which makes them a natural fit for modelling hierarchies of policies.

Options consist of three components: a policy $\pi : \mathcal{S} \times \mathcal{A} \to [0,1]$, a termination condition $\beta : \mathcal{S}^+ \to [0,1]$ (where $+$ denotes the additional inclusion of terminal states) and an initiation set $\mathcal{I} \subseteq \mathcal{S}$. An option $\langle \mathcal{I}, \pi, \beta \rangle$ is available in state s_t if and only if $s_t \in \mathcal{I}$. If the option is taken, the actions are selected according to π until the option terminates stochastically according to β.

According to this definition, primitive actions are simply options which terminate with certainty after one step. As options are composed of primitive actions, any MDP which can be solved using options (selected by a policy-over-options) may also be solved simply with primitive actions. However, a potential advantage of options is that they can be reused to rapidly solve related problems. A classic

[8] an issue further exacerbated by the somewhat arbitrary division of the state space

example of this is the 'four rooms' gridworld problem, in which four rooms are connected by narrow hallways. If, in addition to primitive actions, options are included to move between hallways according to a shortest path, the agent can more rapidly learn to navigate between specific cells in the grid.

The initial focus of research into options focused on learning to choose between options appropriately, as well as how learn about multiple options simultaneously even when following the action choice of just one of them (intra-option learning) (Sutton et al., 1998), and to terminate options when advantageous (interrupting-options) (Sutton et al., 1999). Later work, began to investigate how options might be learned, such as looking for bottleneck states (Stolle and Precup, 2002) or using separate 'problem-space' and 'agent-space' representations (Konidaris and Barto, 2007). Recent research also showed how options policies and termination conditions could be simultaneously learned in a policy gradient framwork (Bacon et al., 2017). However, this work relied on regularisers to prevent agents either learning a single option for each task or switching between options at every step.

2.2.3 Feudal networks

Feudal Networks (FuNs) (Vezhnevets et al., 2017) takes the general inspiration provided by FRL and translates this into a more general deep RL algorithm which can be used to solve a variety of problems. As with FRL, FuN introduces a managerial hierarchy, although experiments were limited to two-level hierarchies of manager and worker.

FuNs was used for problems where the input was high dimensional images, and so the first stage of processing involved using a 'perceptual' convolutional neural network to generate a representation $z_t \in \mathbb{R}^d$ which was supplied as input to both manager and worker. The manager would then transform this into its own internal representation $s_t \in \mathbb{R}^d$ and, using recurrent networks, output a goal $g_t \in \mathbb{R}^d$.

Meanwhile, the recurrent worker takes the same input z_t and produces an embedding vector for every action, represented by rows of the matrix $U_t \in \mathbb{R}^{|a| \times k}$, where $k \ll d$. In the next step, the managerial goals over the last c timesteps are summed by the worker and transformed into a goal-embedding by linearly project-

ing them to generate $w_t \in \mathbb{R}^k$. The worker's policy π is then decided by:

$$\pi_t = \text{SoftMax}(U_t w_t) \tag{2.9}$$

which corresponds to the vector of probabilities over primitive actions.

The linear projection used to generate the goal-embedding w_t is learned by the worker (using gradients coming from its own actions) and so, to ensure that it cannot simply learn to ignore the manager, Vezhnevets et al. (2017) did not allow for any biases which might be used to produce a constant non-zero vector.

The managerial goal g_t not only influences worker behaviour directly but is also used to determine the worker's reward. To compute the intrinsic reward, r_t^I, provided to the worker at time t, the cosine similarity between the d-dimensional goal vector and the change-in-state vector of the agent over a horizon is computed:

$$r_t^I = 1/c \sum_{i=1}^{c} d_{cos}(s_t - s_{t-i}, g_{t-i}) \tag{2.10}$$

where $d_{\cos}(\alpha, \beta) = \alpha^T \beta / (|\alpha||\beta|)$. The use of directional rather than absolute goals is a substantial shift from FRL, and allows for the worker to consistently recieve reward rather than having to wait until the goal has been perfectly achieved or until the time out. The fixed termination condition corresponding to c time steps was treated as a hyperparameter to be optimised.

Crucial to FuNs is the separation between manager and worker. Conventional deep RL approaches typically train in an end-to-end fashion. For the architecture used, this would mean that gradients would backpropagate through the goal output of the manager. However, Vezhnevets et al. (2017) argue that this would deprive managerial goals of any semantic meaning and so they do not allow gradients to propagate from worker to manager, instead treating managerial goals as the actions of an RL agent optimising the task reward.

Whilst FRL followed the principle of reward hiding strictly, FuNs relaxes this, allowing the worker to optimise a weighted sum of the managerially-defined instrinsic reward and the task reward. The weighting is treated as a hyperparameter, α, to

be optimised. Learning for the worker is on-policy and uses advantage actor critic (Mnih et al., 2016):

$$\theta \leftarrow \theta + \Big[R_t + \alpha R_t^I - v_t^w(o_t;\theta)\Big]\nabla_\theta \log \pi(a_t \mid o_t;\theta) \tag{2.11}$$

where R_t and R_t^I correspond to the real and intrinsic return, o_t corresponds to the input image and the worker's value function v_t^w is determined by an internal critic which estimates the value functions for the summed rewards.

The manager in FuN learns according to a form of policy gradient with respect to a model of the worker's behaviour. This assumes that worker state transitions conditioned on the managerial goal will follow a von Mises-Fisher distribution centred on that goal, the satisfaction of which with sufficient training is encouraged by the worker's intrinsic reward. This gives rise to the transition policy gradient for the manager:

$$\theta \leftarrow \theta + \Big[R_t - v_t^m(o_t;\theta)\Big]\nabla_\theta d_{cos}(s_{t+c} - s_t, g_t(\theta)) \tag{2.12}$$

where v_t^m is estimated by the internal critic. The effect of this update is for the manager to move its goals towards directions in which the worker achieves a greater return than expected and away from those in which the worker achieves less return than expected. A final consituent of the FuN architecture includes a 'dilated LSTM' for the manager which operates at a lower temporal resolution, and enables better preservation of memories for long periods than a regular LSTM whilst still learning from every input experience.

Taken together, the FuN architecture allows the application of key ideas from FRL to be more generally applied, albeit in a complex architecture which benefits from substantial hyperparameter tuning (Jaderberg et al., 2017). The experiments also showed strong performance, including on the challenging task of Montezuma's revenge for which an agent must first reach a key to unlock a door before obtaining reward. Pleasingly, the experimenters found that subgoals set by the manager on this task corresponded to reaching the key, as well as other semantically meaningful targets.

2.2.4 Off-policy HRL

Whilst FuNs provided an on-policy policy gradient approach inspired by FRL, subsequent work also developed off-policy methods. In 'HIerarchical Reinforcement learning with Off-policy correction' (HIRO) (Nachum et al., 2018a), as with FuNs, a lower-level controller is supervised with goals that are learned and proposed automatically by a higher-level controller. HIRO uses the off-policy algorithm TD3 (Fujimoto et al., 2018), an improved variant of DDPG, for each controller. Unlike FuNs, goals set by the high-level controller use state observations in their raw forms rather than a learned hidden representation, which the authors found to achieve better performance, albeit on relatively low-dimensional continuous control tasks which formed the basis of their experiments.

Goal-setting in HIRO proceeds as follows: at each time step t, the environment provides an observation state s_t to the higher-level controller which then generates a goal $g_t \in \mathbb{R}^{d_s}$, where d_s is the dimensionality of the observation state. The goal is either sampled from the high-level policy, $g_t \sim \mu^{hi}$, if $t \equiv 0 \pmod{c}$ or otherwise according to a fixed goal transition function $g_t \sim h(s_{t-1}, g_{t-1}, s_t)$. This transition function ensures that a selected goal-vector will continue to point towards a fixed 'target' in the state-space even as the agent moves (for the $c-1$ steps after the goal is initially sampled from μ^{hi}). The goal transition is thus defined as:

$$h(s_t, g_t, s_{t+1}) = s_t + g_t - s_{t+1} \tag{2.13}$$

The intrinsic reward received at each timestep is determined by the distance of the current observation from the target state:

$$r(s_t, g_t, s_{t+1}) = -||s_t + g_t - s_{t+1}||_2 \tag{2.14}$$

To increase reliability and effectiveness, HIRO attempts to address the issue of learning in lower levels altering the effect of actions (goals) in the higher level. To achieve this, an off-policy correction is used which re-labels experiences in the past (stored in the replay buffer) with a high-level action chosen to maximise the prob-

ability of the past lower-level actions. Rather than add a tuple (s_t, g_t, $\sum_c R_{t:t+c-1}$, s_{t+c}) to the replay buffer of the higher level controller, which includes the actual goal used at the time, HIRO replaces this goal with $\tilde{g}_t$ which maximises the probability $\mu^{lo}(a_{t:t+c-1} \mid s_{t:t+c-1}, \tilde{g}_{t:t+c-1})$, where the intermediate goals are computed using the fixed goal transition function h. This optimisation is done approximately by computing this quantity for a number of sampled goals $\tilde{g}_t$ near to the observed state transition and choosing the argmax.

Another approach which used off-policy learning for HRL is 'Hierarchical Actor Critic' (HAC) (Levy et al., 2018). This method shares many similarities to HIRO but does not introduce goal relabelling to adjust for learning in the lower-level policy. It does however introduce a mechanism for Hindsight Experience Replay (Andrychowicz et al., 2017), which enables the lower level controller to learn even when it reaches states far from the goal state. This works by sampling a set of additional goals, substituting them for the true goal state and adding the additional tuples to the replay buffer for further training. Additional goals sampled are generally related to the actual observed trajectory on that episode (e.g. states subsequently visited), providing a useful reward signal for learning when ordinarily goal achievement is rare. This idea is closely linked to general or universal value functions (Sutton et al., 2011; Schaul et al., 2015), which seek to treat all states as potential goals for which the agent would like to accurately predict values. Schaul et al. (2015) used one neural network to represent knowledge about all subgoals, such that values could be predicted for any input start and goal state.

Both HIRO and HAC increase the efficiency of off-policy training by asking the question: what goal could the manager have set to best reflect the actually achieved state change? In the case of HIRO, the surrogate goal which (approximately) answers this question is used to adjust the manager's update whereas in the case of HAC it is used to augment the worker's update.

2.2.5 Multi-agent connections

2.2.5.1 Layered learning

Layered learning (Stone and Veloso, 2000) is a hierarchical multi-agent paradigm which takes a sequential approach to multi-agent training. It aims to address problems where learning a direct mapping from inputs to outputs is intractable, and so involves an overall task being broken down into layers of subtasks. This learning is bottom up, with simplest tasks being learned first and then used as components for learning in the next subtask layer. A learned subtask could contribute to learning in the next layer by constructing a set of training examples, providing input features to the next layer or pruning the output set. For example, in the simulated robotic soccer domain, agents were first trained to intercept the ball. With this skill having been acquired, they then learned to evaluate whether to pass the ball whilst training against other agents with the learned interception skills. In a final stage the robots learned pass selection, using as input their previously learned ability to evaluate passes. Pass evaluation was further used to prune the action space for pass selection. In this way, layered learning achieved much better performance than could be achieved by end-to-end training on simulated robotic soccer. However, the approach in general requires a system designer to specify this task decomposition, which may place unrealistic demands on domain knowledge.

2.2.5.2 Principal-agent problems

Our approach to multi-agent learning involves a manager agent that assign goals to worker agents. In economics, a related problem is known as the 'principal-agent problem' (Jensen and Meckling, 1976) which involves an entity, called the 'principal', which hires another entity, called the 'agent', to perform a task on its behalf. The agent, however, does not have the same objective function as the principal and also has access to different information, such that the principal cannot directly ensure that the agent is acting in its best interest.

Three types of information problems commonly arise; moral hazard, adverse selection and non-verifiability (Laffont and Martimort, 2009). Moral hazard may

occur when the agent takes actions which affect performance but which cannot be easily verified by the principal. Adverse selection can arise when the agent has access to private information relevant to the task which is hidden from the principal. Non-verifiability can occur if there is a disagreement between principal and agent about the state of the world which cannot be easily or cheaply resolved by a third party.

In our work, we introduce manager-worker hierarchies analogous to principal-agent models and so a variety of issues which can arise from this setting, including those involving many agents (Sappington, 1991; Laffont and Martimort, 2009), are relevant. However, as we will see, one advantage of multi-agent RL systems is the possibility to 'program in' incentives to encourage cooperative behaviours. Such manipulation is of course not possible for systems comprised of non-programmable entities, such as humans.

Chapter 3

Feudal multi-agent hierarchies

Many of the results of this chapter were presented in:
Workshop on Structure & Priors for Reinforcement Learning at ICLR 2019
(Ahilan and Dayan, 2019)

3.1 Introduction

We have reviewed two general approaches for addressing multi-agent problems: the decentralised approach can be applied more generally, but may struggle to deal with non-stationarity, whereas the centralised approach addresses this issue but relies on training in simulation and can be difficult to scale. In this chapter, we hope to address the challenges of scalability, non-stationarity and coordination we have described by introducing our new framework of FMH and combining it with decentralised learning.

FMH combines ideas from multi-agent and hierarchical RL, and draws inspiration from real-life managerial hierarchies. In such structures, responsibilities are often divided across members of the team, who agree to be assigned different objectives for the benefit of the collective. For example, members of a company typically have different roles and responsibilities. They will likely report to managers who define their objectives, and they may in turn be able to set objectives to more junior members. At the highest level, the CEO is responsible for the company's overall performance.

In FMH we likewise organise many, simultaneously acting agents into a man-

agerial hierarchy. Instead of each agent optimising a shared reward, we only expose the highest-level manager to this 'task' reward. The manager must therefore learn to optimise this by communicating subgoals, which define a reward function, to the worker agents under its control. Workers learn to satisfy these subgoals by taking actions in the world and/or by setting their own subgoals for workers immediately below them in the hierarchy.

We envisage a benefit from FMH arising from transforming a shared reward problem into one where agents have different goals and can thus more easily assign credit towards their achievement of those goals. If useful behaviours can be learned, a manager agent can then coordinate agents under its control to achieve the overall goal. This has the potential to address the issue of non-stationarity as managerial reward renders the behaviour of workers more predictable.

The diversity of rewards experienced by agents can provide a rich learning signal, but necessarily implies that interactions between agents will not in general be fully cooperative. Nevertheless, through managerial incentives and appropriate managerial behaviour, we hope for agents to achieve collective behaviours which are apparently cooperative, from the perspective of an outside observer viewing performance on the task objective.

Our work builds on FRL and feudal networks, but differs in its inclusion of many agents which act simultaneously. This multipli city enables tasks not only to be divided across time but also across worker agents. Whilst such a division could in theory be applied to single-agent systems, such as the many arms of an octupus, we embrace the explicit multi-agent setting in which observations are not in general shared and communication between agents may be limited.

3.1.1 Hierarchies

We start by defining the types of hierarchical structures allowable in FMH. A critical consideration is ensuring that potentially disastrous feedback cycles for the reward are not possible. This may be achieved by enforcing a directed acyclic graph (DAG) structure. A DAG is a graph in which each edge has an orientation from one vertex to another and for which there are no cycles along any of the defined paths.

We find structures involving only a single manager to be applicable to a range of interesting problems. Whilst we do not investigate automated methods for assigning a particular agent as manager, we note that appropriate assignment of agents to positions in a hierarchy should depend on the varied observation and action capabilities of the agents. Agents exposed to key information associated with global task performance are likely to be suited to the role of manager, whereas agents with more narrow information, but which can act to achieve subgoals are more naturally suited to being workers.

3.1.2 Communication as goals

We outlined an explicit approach to multi-agent communication in Section 2.1.2.2. This divided the action space of agent i into disjoint environment actions $\mathcal{A}_i^e$ and communication actions $\mathcal{A}_i^m$, such that $\mathcal{A}_i^e \cup \mathcal{A}_i^m = \mathcal{A}_i$ and $\mathcal{A}_i^e \cap \mathcal{A}_i^m = \emptyset$. Communication actions did not impact the environmental dynamics or rewards but were directly observed by other agents. In FMH, we also use explicit communication but now allow this communication to alter rewards[1].

The idea of a manager rewarding worker agent(s) spans a wide array of possible approaches. A particularly simple approach would be to allow the manager to directly specify the reward received by workers. In this 'direct reward' approach we might hope that the manager would learn to reward workers when they act in a way useful towards solving the overall task, reinforcing those behaviours in the worker. However, such a free form approach may be difficult to learn; the manager cannot be expected to initially reward the worker in a consistent or useful way and so it may be difficult for the worker to ever learn to reach useful states.

Instead we consider an alternative approach, more similar to methods in hierarchical reinforcement learning, which involves managers specifying goals rather than rewards. The goal is communicated to the worker and defines a reward function for the worker in a fixed way. This approach has the major benefit of making the problem of achieving reward less non-stationary from the perspective of the worker as its rewards are now grounded in its own state space rather than being subject to

[1] the framework of course also allows for ordinary communication which does not alter rewards.

the whims of a managerial administrator. This does however require the designer of the system to specify a mechanism for goal-setting.

Two general approaches can be useful for formalising goal-setting in FMH. In a centralised 'environmental' formalism the reward for agent i, defined as $r_i : \mathcal{S} \times \mathcal{A}_1 \times \ldots \times \mathcal{A}_n \times \mathcal{S} \to \mathbb{R}$, depends on the true state of the environment, the environmental actions of all other agents and the communicated goals of the agent's manager. However, for systems deployed in the real world, agents may not have access to the true environmental state or other agents' actions. We therefore consider a 'worker-computed' formalism where each worker agent computes its own reward locally $r_i : \mathcal{O}_i \times \mathcal{G}_i \times \mathcal{A}_i \times \mathcal{O}_i \to \mathbb{R}$, where $\mathcal{O}_i$ corresponds to the set of all non-goal based local observations, and $\mathcal{G}_i$ corresponds to the set of all possible locally observed goals g_i[2]. We illustrate this 'worker-computed' interpretation in Fig. 3.1.

3.1.3 Communication as control

Specifying a particular agent as manager to control other agents bears an interesting relationship to an approach to cooperative RL mentioned in Section 2.1.2.2, which uses communication to sidestep some of the difficult challenges of multi-agent RL. The idea is that if communication is instantaneous and unrestricted, then each of N agents can communicate their observations to a special agent, and this agent can then communicate back the desired action to each of them. Training with this mechanism is equivalent to single-agent RL, enabling multi-agent issues of non-stationarity and coordination to be addressed. This in turn supports convergence guarantees for decentralised multi-agent training of communicating agents.

Connecting this idea to FMH, we can denote the special agent as a 'manager', responsible for control of other agents in the multi-agent system[3]. However, in real multi-agent systems, communication is not instantaneous and unrestricted, motivating managerial control which is temporally extended and not reliant on micro-management. As we will see, the use of longer-term, reward based goals which

[2] in our implementation communication actions are received on the next time step, so the worker's observed goal in fact corresponds to the manager's selected goal at the previous time step

[3] in this chapter we address problems in which the manager already receives relevant observations and so does not rely on workers to communicate them

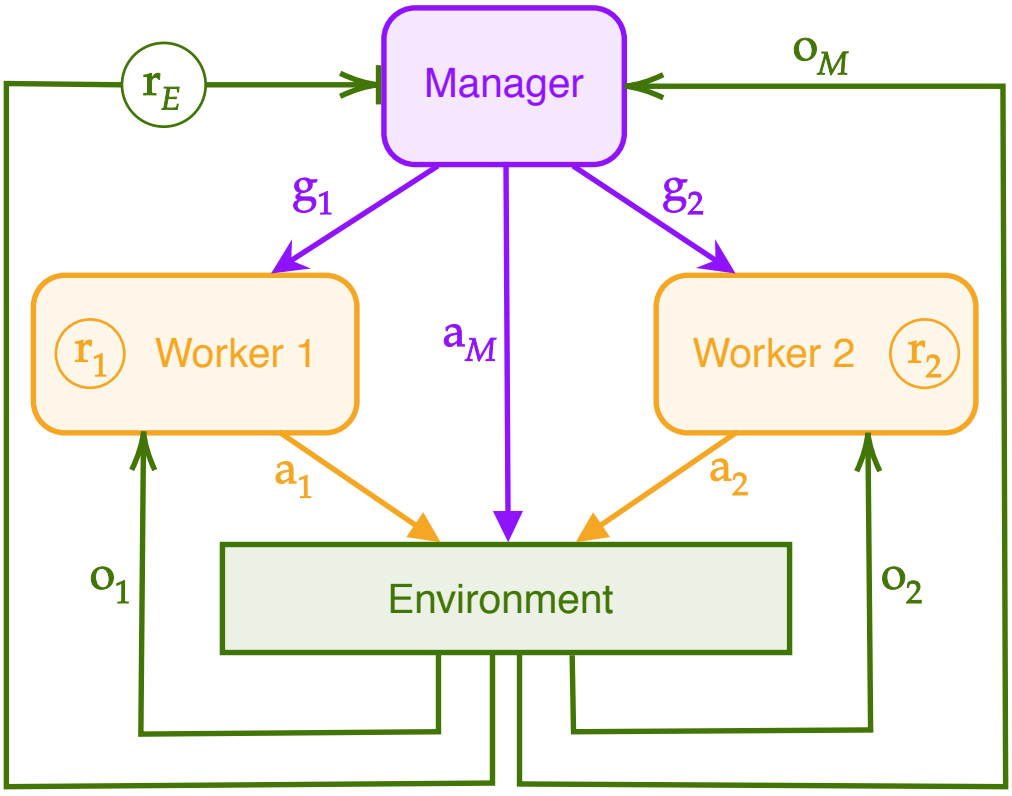

Figure 3.1: The structure of FMH. An example of a worker-computed Feudal Multi-agent Hierarchy with one manager agent and two worker agents. Worker rewards are goal-dependent and computed locally, the manager's reward is provided by the environment.

indirectly control agents, as in hierarchical RL, can be effective, with the resulting temporal abstraction providing additional benefits for scaling to larger numbers of decentralised agents.

3.1.4 Coordination

We can illustrate the ability of a feudal approach to coordinate by considering coordination games, popular in microeconomics, in which there are multiple good and bad Nash equilibria, and it is necessary to find the former. It is intuitively obvious that appointing one of the agents as a manager might resolve the symmetries inherent in a cooperative coordination game in which agents need to take different actions to receive reward.

We consider a game which has two pure strategy Nash equilibria and one mixed

A

Player X \ Player Y	A	B
A	$(0,0)$	$(1,1)$
B	$(1,1)$	$(0,0)$

B

g_A	A	B
A	$(0,1)$	$(1,0)$
B	$(1,1)$	$(0,0)$

C

g_A	A
A	$(0,1)$
B	$(1,1)$

Figure 3.2: Feudal rewards can be used to achieve coordination. (**A**) A coordination game. (**B**) The manager (Player X) communicates goal g_A to the worker (Player Y) such that the worker is only rewarded for taking action A. (**C**) The reduced matrix game allows both agents to coordinate to the Pareto-optimal Nash equilibrium (B,A).

strategy Nash equilibrium which is Pareto dominated by the pure strategies (Figure 3.2A). The challenge of this game is for both agents to choose a single Pareto optimal Nash equilibrium, either (A,B) or (B,A).

For a matrix game, we define the feudal approach as allowing Player X, the manager, to specify the reward player Y will receive for its actions. This is a simplification when compared to the more general setting of a Markov game in which the feudal manager can reward not only actions but also achievement of certain states. In order to specify the reward, we assume that Player X communicates a goal, either g_A or g_B, prior to both players taking their actions. If Player X sends g_A it means that action A is now rewarded for Player Y and action B is not[4]. Player X's rewards are unchanged, and so together this induces the matrix game shown in Figure 3.2B.

Action A for player Y is now strictly dominant and so a rational Player Y will always choose it. By elimination of strictly dominated strategies we therefore find the reduced matrix game of Figure 3.2C.

And so a rational Player X will always choose B, resulting in an overall strategy of (B,A) conditioned on an initial communication of g_A. By symmetry, we can see that conditioned on g_B, rational players X and Y will play (A,B). The feudal approach therefore allows the manager to flexibly coordinate the pair of agents to either Nash equilibrium. For games involving N players, coordination can be achieved by the manager sending out N-1 goals.

[4]we assume the manager can reward the worker even in the case where it itself does not receive reward; there is no conservation of reward.

3.2 Methods

We propose FMH, a framework for multi-agent RL which addresses the three major issues outlined in the introduction: non-stationarity, scalability and coordination.

3.2.1 Discrete actions with Gumbel-Softmax and DDPG

As a framework, FMH can work with many different RL algorithms. For our experiments, we choose to apply it with the single-agent algorithm DDPG, trained in a fully decentralised fashion. We utilise the Gumbel Softmax estimator (Section 1.3.4) for discrete communication actions. Our approach is identical to that of Lowe et al. (2017) who found that DDPG with Gumbel-Softmax (perhaps surprisingly) performed more strongly than other decentralised single-agent RL algorithms such as DQN (Mnih et al., 2015) and Trust-Region Policy Optimisation (Schulman et al., 2015). As we do not experiment with combining FMH with any other algorithms in this chapter, we frequently refer to FMH-DDPG simply as FMH. We provide further details on varieties of algorithms, including those introduced in later chapters, in Section A.2.4.

3.2.2 Goal-setting

In FMH, managers communicate a special kind of message to workers that defines the workers' reward functions according to a specified mapping. As we do not wish the manager to micromanage all aspects of worker behaviour for temporally-extended problems, we explore a scheme in which a manager is able to select between discrete targets which define distance based reward functions. For example, if there are three objects in a room, we might allow the manager to select from three different messages, each defining a worker reward function proportional to the negative distance from a corresponding object. In this context and task, messages therefore correspond to 'goals' requiring approach to different objects. The manager must learn to communicate these goals judiciously in order to solve complex tasks. In turn, the workers must learn how to act in the light of the resulting managerial reward, in addition to immediate rewards (or costs) they might also experience.

Our approach to defining rewards generally uses the 'worker-computed' perspective in which workers are able to compute rewards locally based on their immediate observations. Our choice of reward scheme imposes structure on the goal space making it amenable to discrete goals.

3.2.3 Pretraining and temporally-extended subgoals

We next consider the issue of non-stationarity. This frequently arises in multi-agent RL because the policies of other agents may change in unpredictable ways as they learn. For a Listener agent, this introduces non-stationarity in the relationship between the messages it receives and the reward it obtains. By contrast, in FMH we allow manager agents to determine the reward functions of workers, compelling the workers towards more predictable behaviour from the perspective of the manager. However, the same issue applies at the starting point of learning for workers: they will not yet have learned how to satisfy the goals. We would therefore expect a manager to underestimate the value of the subgoals it selects early in training, potentially leading it sub-optimally to discard subgoals which are harder for the worker to learn.

Thus, it would be beneficial for worker agents already to have learned to fulfill managerial subgoals. We address this issue practically in two steps. First, we introduce a bottom-up 'pretraining' procedure, in which we initially train the workers before training the manager. Although the manager is not trained during this period, it still acts in the multi-agent environment and sets subgoals for the worker agents. As subgoals are initially of (approximately) equal value, the manager will explore them uniformly. If the set of possible subgoals is sufficiently compact, this will enable workers to gain experience of each potential subgoal.

This pretraining does not completely solve the non-stationarity problem. This is because the untrained manager will, with high probability, vacillate between subgoals, preventing the workers under its command from having any reasonable hope of extracting reward. Ideally, we would want worker agents to optimise managerial goals over the longer term, but if the manager continually changes its mind such that the current task has a high probability of terminating soon, this disincentivises

the worker from doing so. One interpretation of vacillation is therefore that it effectively decreases the worker's γ parameter such that it only concerns itself with the immediate reward.

To alleviate this problem, we therefore want managers not only to try out a variety of subgoals but also to *commit* to them long enough that they have any hope of being at least partially achieved. Thus, the second component of the solution is a communication-repeat heuristic for the manager such that goal-setting is temporally extended. Action repeats are commonly used in Atari environments (Mnih et al., 2015; Vezhnevets et al., 2017) but we do not do this for regular actions in the MPE, restricting repeats only to communication; we demonstrate the effectiveness of this in our experiments.

3.2.4 Parameter sharing

We apply our method to scenarios in which a large number of agents have identical properties. For convenience, when training using a decentralised algorithm (FMH-DDPG or vanilla DDPG) we implement parameter sharing among identical agents, in order to train them efficiently. We only add experience from a single agent (among those sharing parameters) into the shared replay buffer, and carry out a single set of updates. We find this gives very similar results to training without parameter sharing, in which each agent is updated using its own experiences stored in its own replay buffer (see Appendix A.1.2), but requires fewer samples.

3.3 Experimental results

We used the multi-agent particle environment (MPE) described in the introduction to Part I. We train RL agents which have both an actor and a critic, each corresponding to a feedforward neural network. We give a detailed summary of all hyperparameters used in Appendix A.1.1.

3.3.1 Cooperative communication

We first experiment with an environment implemented in Lowe et al. (2017) called 'Cooperative Communication' (Figure 3.3), involving a Speaker and Listener, which we described in the introduction to Part I. In the original problem the

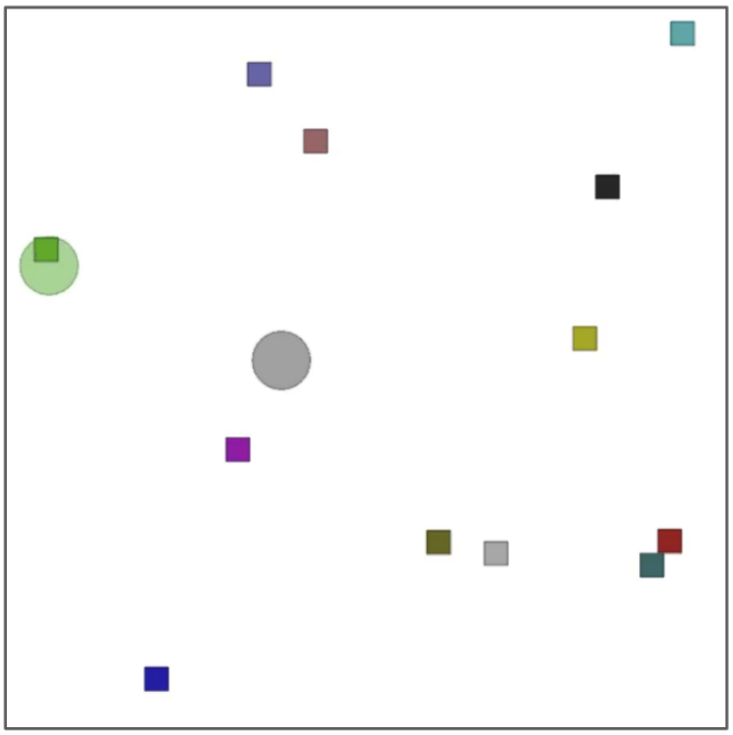

Figure 3.3: Cooperative Communication. The Speaker (grey circle) sees the colour of the Listener's target, which indicates that the green square is the target on this episode. It communicates a message to the listener at every time step. For illustration we colour the Listener according to the colour of its target (although it cannot see this colour). Here there are 12 landmarks and the agent trained using FMH has successfully navigated to the correct landmark.

Listener must navigate to a randomly selected landmark and is rewarded proportional to the negative square distance from the target landmark. By convention we instead use the negative distance for all experiments, as we found this performed better for all algorithms. We also note that, although reward is provided during the episode, it is used only for training agents and not as an observation that could be used for servoing.

Although this task might seem simple, it is actually challenging for many RL methods. Lowe et al. (2017) trained, in a decentralised fashion, a variety of single-agent algorithms, including DDPG, DQN and trust-region policy optimisation Schulman et al. (2015) on a version of this problem with three landmarks and demonstrated that they all perform poorly on this task. Of these methods, DDPG reached the highest level of performance and so we use DDPG as the strongest commonly used baseline for the decentralised approach. We also compare our results

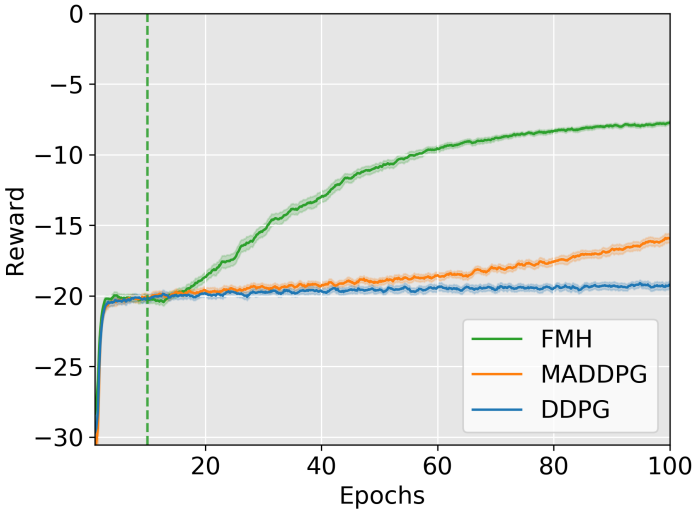

Figure 3.4: Training on Cooperative Communication. FMH substantially outperforms MADDPG and DDPG in a task with 12 possible landmarks. The dashed green line indicates the end of pretraining for FMH.

to MADDPG, which combines DDPG with centralised training. MADDPG was found to perform strongly on Cooperative Communication with three landmarks, far exceeding the performance of DDPG.

For FMH, we also utilised DDPG, but reverted to the more generalizable decentralized training that was previously ineffective[5]. Crucially, we assigned the speaker the role of manager and the listener the role of worker. The speaker can therefore communicate messages which correspond to the subgoals of the different coloured landmarks. The listener is not therefore rewarded for going to the correct target but is instead rewarded proportional to the negative distance from the speaker-assigned target. The speaker itself is rewarded according to the original problem definition, the negative distance of the listener from the true target.

We investigated in detail a version of Cooperative Communication with 12 possible landmarks (Figure 3.3). FMH performed significantly better than both MADDPG and DDPG (Figure 3.4) over a training period of 100 epochs (each epoch corresponds to 1000 episodes). For FMH, we pretrained the worker for 10 epochs and enforced extended communication over 8 time steps (each episode is 25 time steps).

[5]we consider a centralised approach to FMH in the next chapter

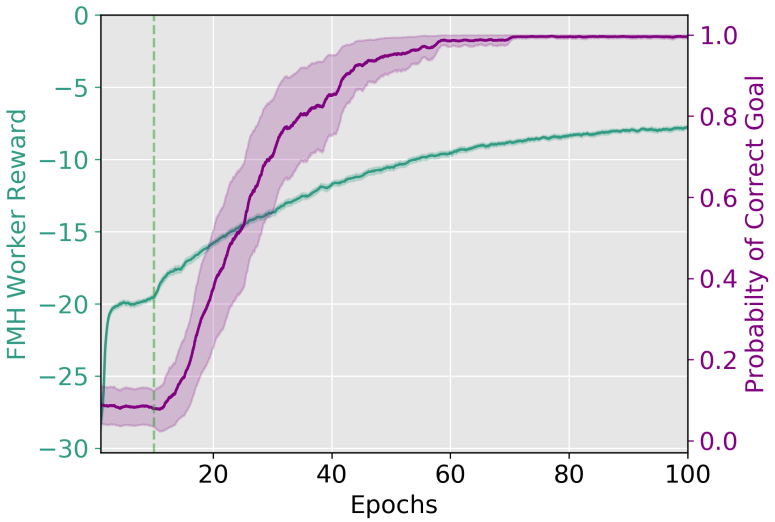

Figure 3.5: Analysing Cooperative Communication. FMH worker reward and the probability of the manager correctly assigning the correct target to the worker. The manager learns to assign the target correctly with probability 1.

In Figure 3.5, the left axis shows the reward received by the FMH worker (listener) over training. This increased during pretraining and again immediately after pretraining concludes. Managerial learning after pretraining resulted in decreased entropy of communication over the duration of an episode (Figure 3.6), which is desired due to the fixed nature of the target. This decrease occured from a relatively low entropy starting point due to our communication repeat heuristic for FMH.

The further reduction in entropy after pretraining allowed the worker to optimise the managerial objective more effectively. This in turn enabled the manager to assign goals correctly, with the rise in the probability of correct assignment occurring shortly thereafter (Figure 3.5; right axis), then reaching perfection.

Our results show how FMH resolves the issue of non-stationarity. Initially, workers are able to achieve reward by learning to optimise managerial objectives, even whilst the manager itself is not competent. This learning elicits robust behaviour from the worker, conditioned on managerial communication, which makes workers more predictable from the perspective of the manager. This then enables the manager to achieve competency – learning to assign goals so as to solve the overall task.

Our implementation of FMH used both pretraining and extended goal-setting

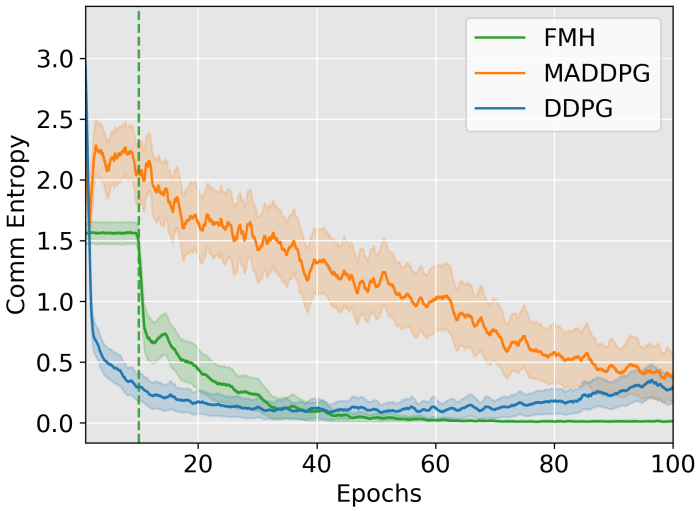

Figure 3.6: Entropy (base 2) of managerial communication. Entropy over the duration of an episode at different stages in training.

with a communication repeat heuristic. Pretraining the worker improved performance on this task (Figure 3.7, with extended communication (goal-setting) for 8 time steps), although even without pretraining FMH still performed better than MADDPG and DDPG.

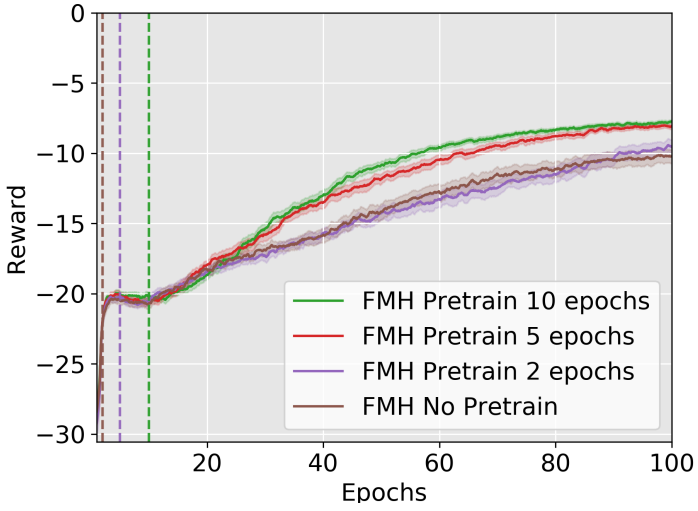

Figure 3.7: Pretraining in FMH. Pretraining, although not essential, supports faster learning

The introduction of extended communication is a more critical addition than

Number of		Final Reward				Epochs until Convergence		
Listeners	Landmarks	FMH	MADDPG	DDPG	CoM	FMH	MADDPG	DDPG
1	3	$\mathbf{-6.63 \pm 0.05}$	$\mathbf{-6.58 \pm 0.03}$	-14.26 ± 0.07	-17.28	56	24	55
1	6	-6.91 ± 0.07	$\mathbf{-6.69 \pm 0.06}$	-18.10 ± 0.07	-18.95	57	66	42
1	12	$\mathbf{-7.79 \pm 0.06}$	-15.96 ± 0.09	-19.32 ± 0.11	-19.56	–	–	36
3	6	$\mathbf{-7.10 \pm 0.04}$	-11.13 ± 0.03	-18.90 ± 0.05	-18.95	77	–	50
5	6	$\mathbf{-7.17 \pm 0.03}$	-18.47 ± 0.04	-19.73 ± 0.06	-18.95	79	75	53
10	6	$\mathbf{-8.96 \pm 0.03}$	-19.80 ± 0.06	-21.19 ± 0.03	-18.95	–	59	32

Table 3.1: Cooperative Communication with varying conditions. Performance of FMH, MADDPG and DDPG with different numbers of Listeners and landmarks. Final reward is determined by training for 100 epochs and evaluating the mean reward per episode in the final epoch. We indicate no convergence with a $-$ symbol. For further details see Suppl. Mat. A.1.3.

pretraining. In Figure 3.8 we show the performance of FMH with goal-setting over various number of time steps (and fixed pretraining period of 10 epochs). When there was no communication repeat (Comm 1), performance was similar to MADDPG, but introducing even a single repeat greatly improved performance. By contrast, introducing communication repeats to MADDPG, which does not treat communication as goals, did not improve performance (Figure 3.9).

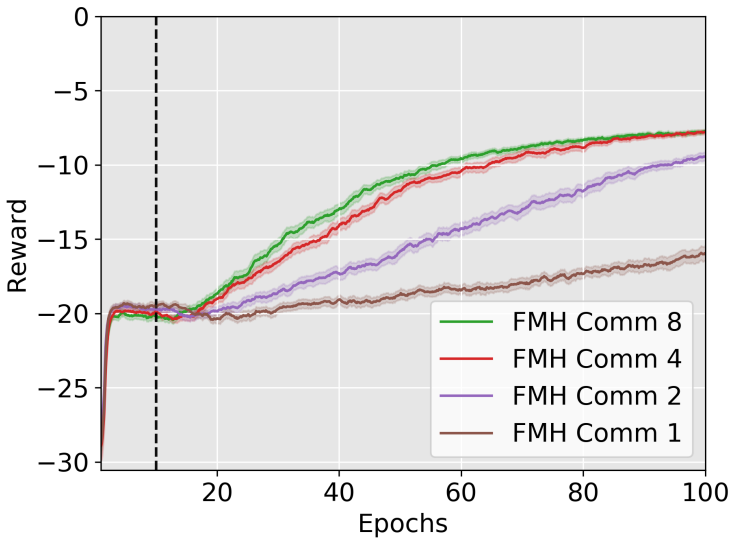

Figure 3.8: Extended communication in FMH. Extended communication and thus goal-setting improves performance for FMH.

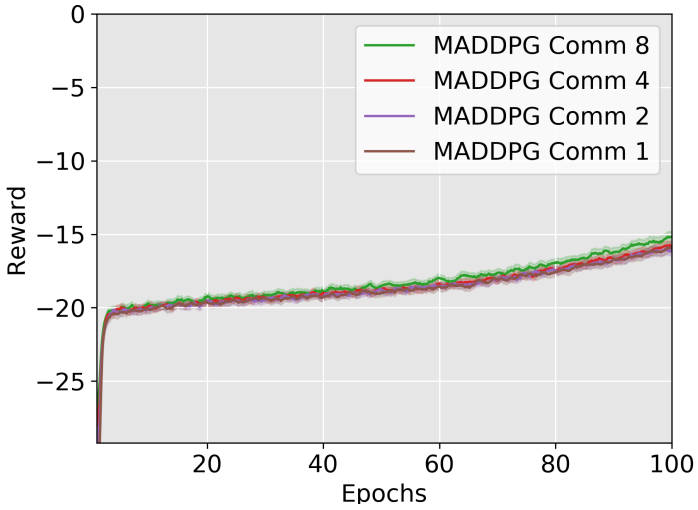

Figure 3.9: Extended communication in MADDPG. Extended communication does not significantly improve the performance of MADDPG.

3.3.2 Scaling to many agents

We next scaled Cooperative Communication to include many listener agents. At the beginning of an episode each listener is randomly assigned a target out of all the possible landmarks and the colour of each listener's target is observed by the speaker. The speaker communicates a single message to each listener at every time step. To allow for easy comparison with versions of Cooperative Communication with only one listener, we normalised the reward by the number of listeners. As discussed in the methods, we shared parameters across listeners for FMH and DDPG.

In Table 3.1 we show the performance of FMH, MADDPG and DDPG for variants of Cooperative Communication with different numbers of listener agents and landmarks. Consider the version with 6 landmarks, which we found that MADDPG could solve with a single listener within 100 epochs (unlike for 12 landmarks). On increasing the number of listeners up to a maximum of 10, we found that FMH scales much better than MADDPG; FMH was able to learn an effective policy with 5 listener agents whereas MADDPG could not. Further, FMH even scaled to 10 listeners, although it did not converge over the 100 epochs.

To aid interpretation of the reward values in Table 3.1 we also compare per-

formance to a policy which simply moves to the centroid of the landmarks. This 'Centre of Mass' (CoM) agent was trained using MADDPG until convergence on a synthetic task in which reaching the centroid is maximally rewarded, and then evaluated on the true version of the task. We find that both MADDPG and DDPG do not perform better than the CoM agent when there are 10 listeners and 6 landmarks.

In Figure 3.10 we show the final state achieved on example episodes of this version of the task, for agents trained using MADDPG and FMH. After training for 100 epochs, MADDPG listeners do not find the correct targets by the end of the episode whereas FMH listeners manage to do so.

3.3.3 Cooperative coordination

We then designed a task to test coordination called 'Cooperative Coordination' (Figure 3.11). In this task, there are 6 landmarks. At the beginning of each episode, three landmarks are randomly selected to be green targets and 3 blue decoys. A team of 3 agents must navigate to cover the green targets whilst ignoring the blue decoys, but they are unable to see the colours of the landmarks. A fourth agent, the speaker, can see the colours of the landmarks and can send messages to the other agents (but cannot move). All agents can see each others' positions and velocities, are large

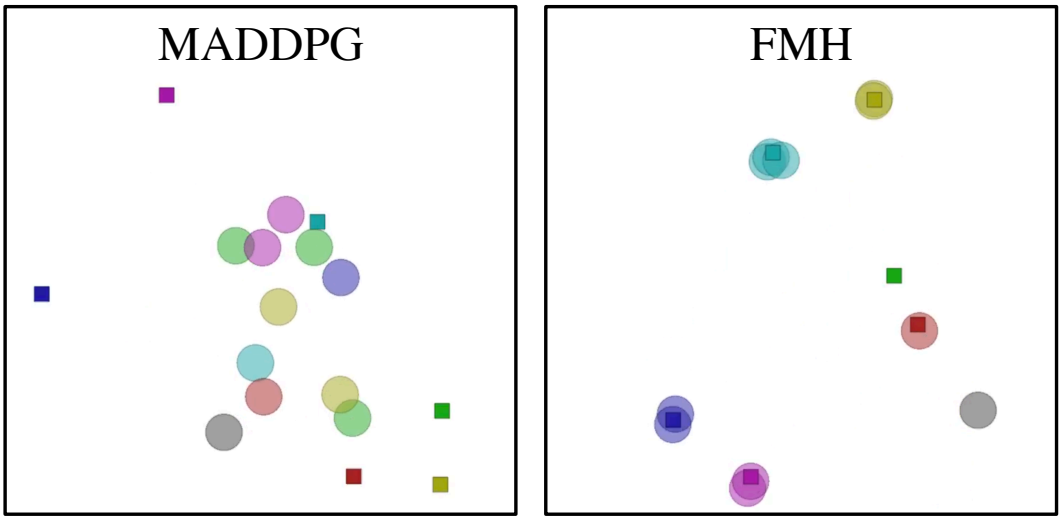

Figure 3.10: Scaling Cooperative Communication. 10 listeners with 6 landmarks - final time step on example episodes for MADDPG and FMH. For FMH the Listener agents navigate to their correct targets whereas for MADDPG they do not.

in size and face penalties if they collide with each other. The task shares aspects with 'Cooperative Navigation' from Lowe et al. (2017) but is considerably more complex due to the greater number of potential targets and the hidden information.

We apply FMH to this problem, assigning the speaker agent the role of manager. One consideration is whether the manager, the worker, or both should receive the negative penalties due to worker collisions. Here we focus on the case in which the manager only concerns itself with the 'task' reward function. Penalties associated with collisions are therefore experienced only by the workers themselves, which seek to avoid these whilst still optimising the managerial objective.

In Figure 3.12 we compare the performance of FMH to MADDPG and DDPG. As with Cooperative Communication, FMH does considerably better than both after training for 150 epochs. This is further demonstrated when we evaluate the trained policies over a period of 10 epochs: the left side of Figure 3.13 shows the mean

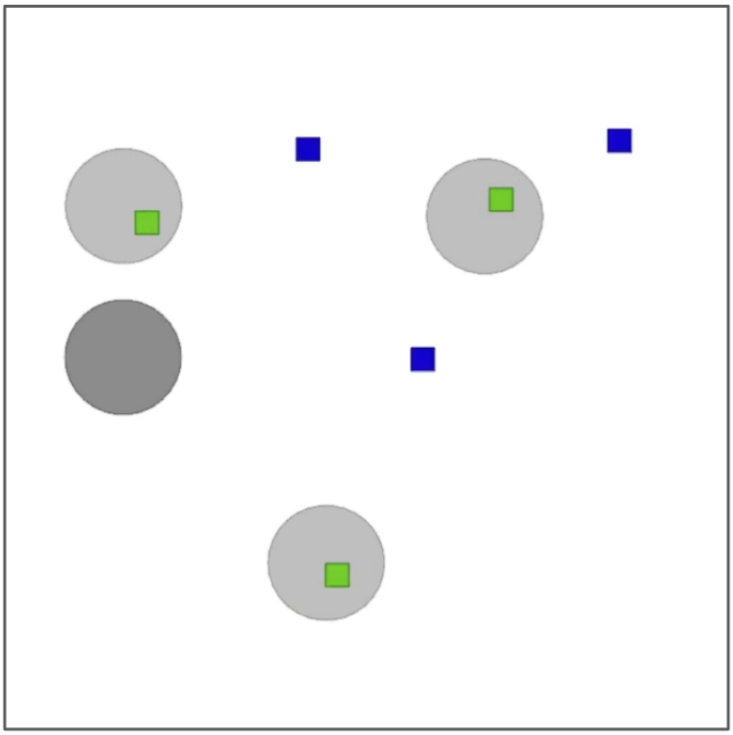

Figure 3.11: Cooperative Coordination. Three listeners (light grey agents) must move to cover the green landmarks whilst ignoring the blue landmarks. However, only the speaker (dark grey agent) can see the landmarks' colours; it communicates with the listeners at every time step. In this example, FMH agents have successfully coordinated to cover the three correct targets.

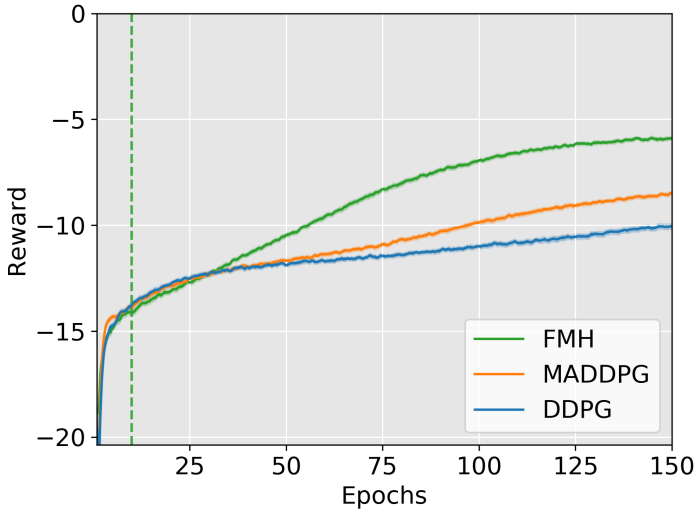

Figure 3.12: Training on Cooperative Coordination. FMH performs significantly better than MADDPG and DDPG. The dotted green line indicates the end of pre-training for FMH

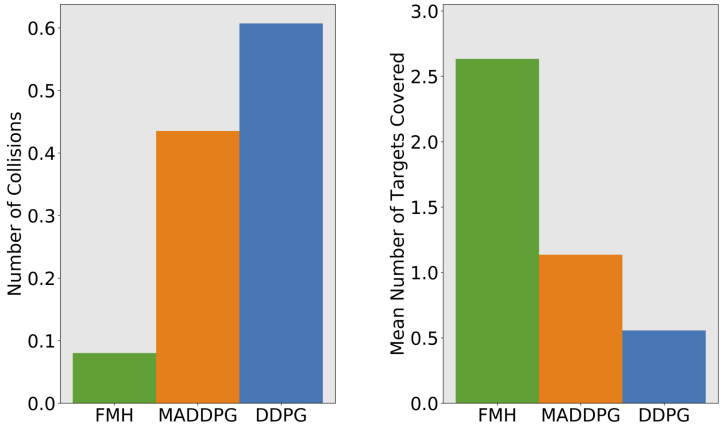

Figure 3.13: Evaluating Cooperative Coordination. **Left**: Agents trained using FMH cover on average more targets, by the end of the episode, than MADDPG and DDPG. **Right**: Agents trained using FMH avoid collisions more effectively than MADDPG and DDPG over the duration of an episode.

number of targets covered by the final time step of the episode, for which FMH more than doubles MADDPG. The right side of Figure 3.13 shows the mean number of collisions (which are negatively rewarded) during an episode. FMH collides very rarely whereas MADDPG and DDPG collide over 4 times more frequently.

We also implement a version of Cooperative Coordination in which the manager is responsible not only for coordinating its workers but must also navigate to targets itself. This involved 2 listeners and 1 speaker which together need to cover

the 3 green targets. We find that the manager learns to do this, outperforming both MADDPG and DDPG (Figure 3.14). Whilst this result is not surprising given our previous findings, it illustrates the point that in FMH managers are simply agents, and so may take primitive actions in a multi-agent environment, whereas in the original feudal RL they do not take primitive actions and are responsible only for goal-setting.

3.3.4 Exploiting diversity

One role of a manager is to use the diversity it has available in its workers to solve tasks more effectively. We tested this in a version of Cooperative Coordination in which one of the listener agents was half the mass of the other two and so could reach farther targets more quickly.

We trained FMH (without parameter sharing due to worker diversity) on Cooperative Coordination and then evaluated the trained policies on a 'Two-Near, One-Far' (TNOF) version of the task in which one target is far away and the remaining two are close. In this task, the agents start at the bottom of the environment. Two green targets are located nearby (in bottom 40 percent of screen) whereas one target is far away (in top 30 percent of screen). The x-coordinates are randomly sampled

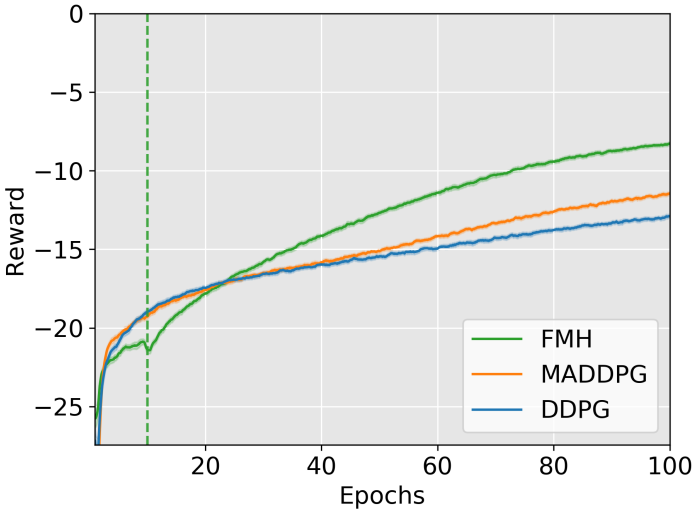

Figure 3.14: Mobile Manager. FMH performs well, even when the manager is required to move to cover targets whilst also setting goals for workers

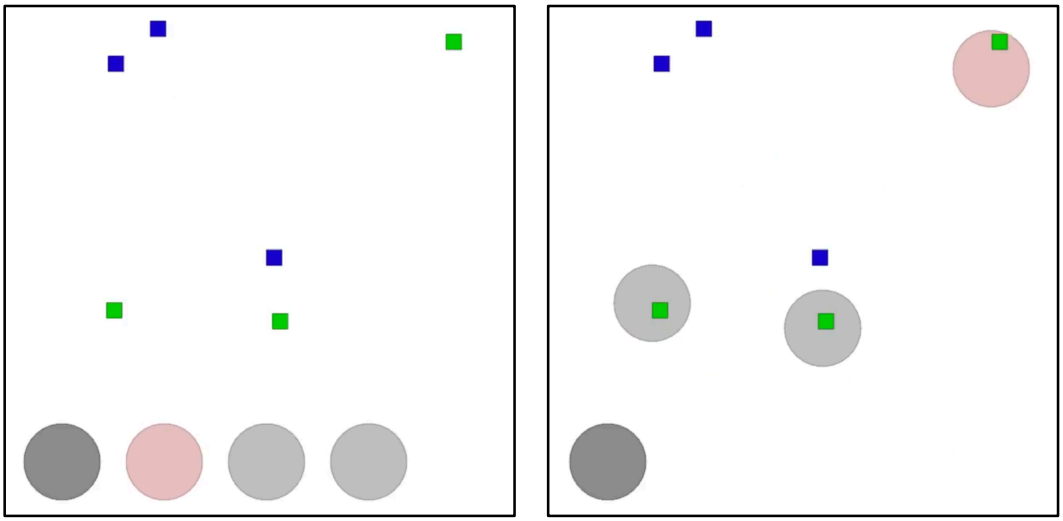

Figure 3.15: Two-Near, One-Far task. FMH solves the TNOF task (example episode). **Left**: Agents are initialised at the bottom of the environment, two targets are close by, and one far away. **Right**: By the end of the episode, the faster (red) agent covers the farther target on the right, despite starting on the left.

at the beginning of each episode and blue decoys are also added (one nearby, two far).

Our choice of this evaluation task was to investigate whether the manager, which was trained on the general problem, had learned the specific strategy of assigning the farthest target to the fastest agent. We found this to be true 100 percent of the time (evaluating over 10 epochs); we illustrate this behaviour in Figure 3.15.

3.3.5 Conclusion

We have introduced our general framework of FMH and applied it in the decentralised case to a range of cooperative multi-agent problems. We found that, given an adequate set of subgoals from which to choose, our approach performs, and particularly scales, substantially better than alternative approaches that use shared rewards. We discuss these results and future work in Chapter 5 but next turn to centralised training and its combination with FMH.

Chapter 4

Centralised policy actor-critic

4.1 Introduction

The FMH method that we discussed in the previous chapter solves cooperative communication problems in a completely decentralized manner, applying the DDPG algorithm to each RL agent. This allows it to work in the various situations in which one agent does not require direct access to the observations of other agents, a frequent fate for real RL agents whose training happens in the real world.

However, it is also very common in the RL community to start (and sometimes end) with simulated environments. Not only can this help reduce the potential for possibly catastrophic errors, such as the collision of a driverless car, but it can also make learning vastly cheaper and faster. Making learning faster can help compensate for the poor sample efficiency of model-free approaches, allowing them to achieve unprecedented performance on complex simulated tasks.

Access to a simulator has been exploited in an additional manner in a multi-agent setting - enabling the centralized training of ultimately decentralized policies (Section 2.1.3.2). Ultimately, to be deployed, agents need to work within their own informational limitations; however, during training it is possible to provide them with extra information, such as the observations and actions of all other agents. This can enable the convergence of decentralized policies to a local optimum to be guaranteed under strict conditions (Konda and Tsitsiklis, 2000; Foerster et al., 2018).

In this chapter we explore ideas which exploit the extra observations available in simulation for centralised training. We begin our exploration by examining how centralised value functions may be combined with FMH, and consider a hybrid approach in which the manager is centralised but the workers are decentralised. In general, we expect each worker to concern itself primarily with the task it has been assigned, perhaps with a few local interactions, whereas we expect the manager to have a broader view of the aggregate behaviour of its workers. An interesting hypothesis is therefore that such a hybrid system would learn more effectively on multi-agent problems by striking an effective balance between the potential scalability of decentralised approaches and the ability of centralised training to alleviate non-stationarities in each agents perceived environment.

We next consider extending centralisation beyond critics to also include policies, and introduce a Centralised Policy Actor-Critic (CPAC), an actor-critic with both a centralised policy and a centralised critic. We define a centralised policy as one which conditions its actions on the observations of all agents, unlike a decentralised policy which only receives the agent's immediate observations. For the fully-cooperative case, we introduce a Single-agent CPAC, responsible for learning to control the joint actions of all agents in the multi-agent system. As it does not need to learn to communicate between agents, we find that it can solve tasks in which information is distributed across agents, which ordinary multi-agent methods find difficult. As we still desire decentralised multi-agent policies, we then train these in a second phase, using the CPAC's critic to guide learning by introducing an alteration to the conventional multi-agent policy gradient.

We can also apply this general idea to FMH. For instance, consider the case in which workers must be assigned to gather information on behalf of the manager and must learn to communicate this. This is problematic for our worker-computed formulation of FMH because the feudal reward structure provides no incentive for workers to communicate relevant state information to the manager. This is because the workers' rewards are grounded entirely in their observation space. Whilst approaches which mix managerial and worker reward may be effective (Vezhnevets

et al., 2017), it is perhaps unclear how a manager would learn to assign workers information-gathering tasks if this relies on workers also communicating this information in a useful way.

As with the single-agent CPAC, we can introduce a problem decomposition and two phases of training to address the problem of learning. Our two phases are are motivated by the Communication as Control algorithm (Section 3.1.3) in which-workers communicate their observations to the manager, and then the manager uses the aggregated information to control the workers. We tackle this in reverse by first training our hierarchy whilst (by exploiting centralization) providing the manager with the observations of its workers, enabling it to learn to assign goals correctly. We then train the multi-agent system to recover this behaviour when these observations are no longer directly available to the manager and so the relevant information must be communicated by the workers. We assume a bottleneck in communication such that workers observations cannot simply be relayed.

Our first phase uses a CPAC for the manager, such that it can see the state of its workers. Unlike the cooperative case, the feudal CPAC controls only the manager's actions, with workers themselves using ordinary MACs. By providing the manager with the CPAC, we allow it to directly set goals for workers using their observations without relying on them communicating the relevant aspects. However, as a centralised policy for the manager requires information that is not available outside simulation, a second phase of training is required. In this phase we incentivise the manager and workers to recapitulate their behaviours from the previous phase, but with the workers additionally learning to communicate the relevant information to the manager, which now learns a decentralised policy. To achieve this we train using the cooperative task objective, but use the critics trained in the first stage to guide learning of the new policy. We use a modified multi-agent policy gradient, similar to the one used for the single-agent CPAC. We find that our two phases finesse the complexities of training in FMH, enabling workers to obey the commands of the manager whilst also communicating effectively.

4.2 Methods

4.2.1 Feudal MADDPG

FMH can be applied to centralised as well as decentralised methods. To explore the effectiveness of centralisation we begin by combining it with MADDPG, using a differing implementation from that of Lowe et al. (2017) introduced by Iqbal and Sha (2019). We detail these differences in the Appendix A.1.5. Our initial experiments with MADDPG find it effective at learning even without extended communication and pretraining (Appendix A.1.4), and so for simplicity we no longer enforce these. As before we also utilise parameter sharing when possible amongst worker agents, which greatly reduces training time.

Feudal MADDPG provides each agent with a centralised value function, but it is plausible that not all agents would benefit from this. In our first experiment we test a hybrid approach in which the manager has a centralised value function but the workers do not. The manager is therefore a MADDPG agent whereas the workers are DDPG agents.

4.2.2 Single-agent CPAC

We can train a CPAC to optimise the task objective by outputting the actions of all agents (Figure 4.1). This is is equivalent to direct application of single agent RL with a centralised controller to the MARL problem. Single-agent learning may be more effective at solving these problems than equivalent multi-agent approaches because coordination between agents may be easier and communication between agents need not be learned.

We denote policies trained in the first stage as π and policies trained in the second stage as μ. In Phase I of training, the Single-agent CPAC is treated as a DDPG agent (Section 1.2.7) with determinstic policy $\pi(\boldsymbol{o};\theta)$, but in general policies may be either deterministic or stochastic. The policy and critic receive as input the observation of all agents $\boldsymbol{o}$, and the critic additionally receives the actions of all agents $\boldsymbol{a}$. The policy gradient is therefore:

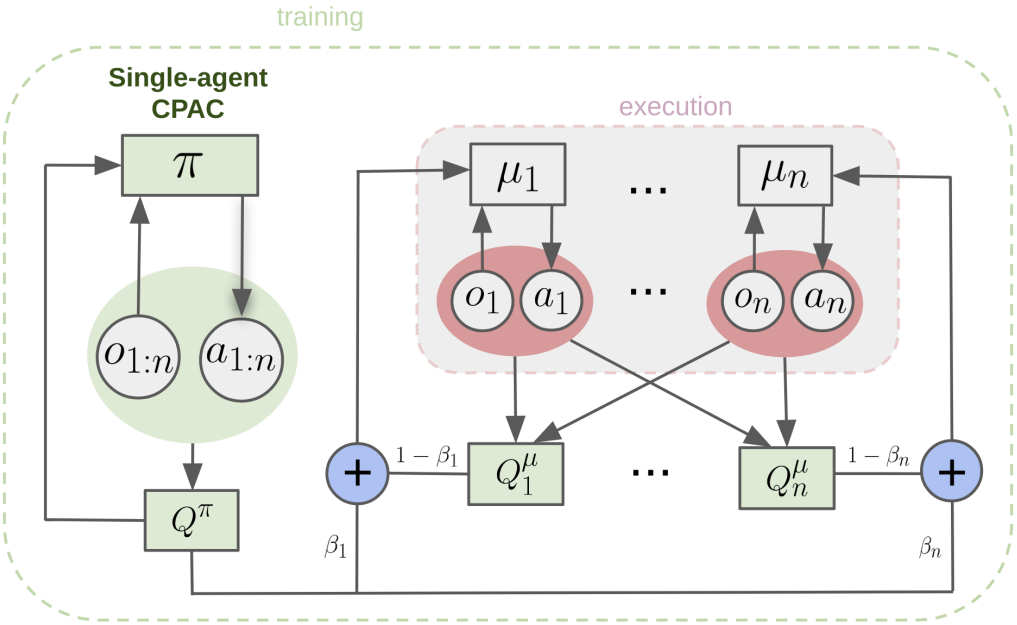

Figure 4.1: Single-agent Centralised Policy Actor-Critic. A single-agent CPAC (left) can be used to train a MAC (right). The CPAC policy π takes in the observations of all agents and outputs the actions for all agents (without communication actions). Its critic receives the observations and actions of all agents. The MAC policy is updated using a convex combination of the value of its own centralised critic and the CPAC critic, with a weighting of β_i which can depend on the agent.

$$\nabla_\theta J(\theta) = \mathbb{E}_{\boldsymbol{o}\sim\mathcal{D},\boldsymbol{a}\sim\boldsymbol{\pi}}[\nabla_\theta \pi(\boldsymbol{o}) \nabla_a Q^{\boldsymbol{\pi}}(\boldsymbol{o},\boldsymbol{a})|_{a=\pi(\boldsymbol{o})}]. \tag{4.1}$$

As communication is irrelevant in this phase, we remove communication actions, which sets the corresponding communication observations uniformly to zero. In Phase II we use the learned single-agent critic to update the multi-agent MADDPG policies by introducing a new policy gradient objective:

$$\nabla_{\phi_i} J(\phi_i) = \mathbb{E}_{\boldsymbol{o}\sim\mathcal{D},\boldsymbol{a}\sim\boldsymbol{\mu}}[\nabla_{\phi_i}\mu_i(o_i)\nabla_{a_i}(\beta_i Q^{\boldsymbol{\pi}}(\boldsymbol{o},\boldsymbol{a}) + (1-\beta_i)Q_i^{\boldsymbol{\mu}}(\boldsymbol{o},\boldsymbol{a}))|_{a_i=\mu_i(o_i)}] \tag{4.2}$$

for each agent. Here β_i is a parameter between 0 and 1 and ϕ_i are the parameters of the i'th decentralised actor μ_i. We set the communication observations to

zero before inputting them to $Q^{\boldsymbol{\pi}}$ to ensure consistency with the previous stage. We do not further train $Q^{\boldsymbol{\pi}}$, which we have learned in Phase I, but do train $Q_i^{\boldsymbol{\mu}}$ (approximated by Q_i^v) according to:

$$\mathcal{L}(v_i) = \mathbb{E}_{\boldsymbol{o},\boldsymbol{o'}\sim\mathcal{D},\boldsymbol{a},\boldsymbol{a'}\sim\boldsymbol{\mu}}[(Q_i^v(\boldsymbol{o},\boldsymbol{a}) - y)^2] \tag{4.3}$$

where the target $y = r + \gamma Q_i^v(\boldsymbol{o'},\boldsymbol{a'})$ uses the shared task reward.

The parameters β_i for the multi-agent policy gradient are interesting to examine. In the special case where β_i are all zero we recover the conventional multi-agent policy gradient, which does not take guidance from the first stage, including the managerial CPAC, at all. This is therefore equivalent to simply training MADDPG on the task reward. In the case where all $\beta_i = 1$ each decentralised policy μ_i is simply concerned with optimising the single-agent critic from the first stage Q^{π}. When β_i are between zero and one each agent seeks to optimise a mixture of both. As we would like agents to both learn to mimic single-agent behaviours learned in the first stage and additionally learn communication, we choose a balance of these two incentives, keeping β_i fixed at 0.5 for all agents. If we were particularly concerned about the ability of our approach to converge to a local optimum, we could also try an annealing approach where beta is gradually decayed from 1 to 0 for each agent. This recovers the convergence guarantees of conventional MAC training, whilst potentially finding better local optima. However, we do not find this necessary for our experiment to succeed.

4.2.3 Feudal CPAC

We wish to train FMH on problems in which the manager does not have direct access to the information it needs for correct target assignment but for which workers can acquire such information. We use MADDPG as an example, but our approach is general for any multi-agent actor-critic (MAC) which allows for each agent to have a separate critic.

For convenience we label the manager as agent 0 of n agents, with the rest being workers. In Phase I of training, the manager's CPAC is trained with conven-

tional MADDPG updates, but with deterministic policy $\pi_0(\boldsymbol{o};\theta_0)$ conditioned on the observations of all agents:

$$\nabla_{\theta_0} J(\theta_0) = \mathbb{E}_{\boldsymbol{o}\sim\mathcal{D},\boldsymbol{a}\sim\boldsymbol{\pi}}[\nabla_{\theta_0}\pi_0(\boldsymbol{o})\nabla_{a_0} Q_0^{\boldsymbol{\pi}}(\boldsymbol{o},\boldsymbol{a})|_{a_0=\pi_0(\boldsymbol{o})}]. \tag{4.4}$$

In addition to training the manager we also train workers using the policy gradient, but with decentralised policies and centralised critics, as in conventional MADDPG. The centralised critics for each agent $Q_i^{\boldsymbol{\pi}}$ (approximated by Q_i^w), including the manager, minimise the objective:

$$\mathcal{L}(w_i) = \mathbb{E}_{\boldsymbol{o},\boldsymbol{o'}\sim\mathcal{D},\boldsymbol{a},\boldsymbol{a'}\sim\boldsymbol{\pi}}[(Q_i^w(\boldsymbol{o},\boldsymbol{a}) - y)^2] \tag{4.5}$$

where $y = r_i^F + \gamma Q_i^w(\boldsymbol{o'},\boldsymbol{a'})$. Here r_i^F corresponds to FMH rewards, with workers receiving feudal rewards and the manager receiving the task reward $r_0^F = r$.

Having trained our critics for each agent in Phase I, we wish to use them to guide learning of a MAC with decentralised policies $\boldsymbol{\mu}$ for each agent in Phase II. Similar to Equation 4.2 we use the following policy gradient objective:

$$\nabla_{\phi_i} J(\phi_i) = \mathbb{E}_{\boldsymbol{o}\sim\mathcal{D},\boldsymbol{a}\sim\boldsymbol{\mu}}[\nabla_{\phi_i}\mu_i(o_i)\nabla_{a_i}(\beta_i Q_i^{\boldsymbol{\pi}}(\boldsymbol{o},\boldsymbol{a}) + (1-\beta_i)Q_i^{\boldsymbol{\mu}}(\boldsymbol{o},\boldsymbol{a}))|_{a_i=\mu_i(o_i)}] \tag{4.6}$$

for each agent. Here β_i is a parameter between 0 and 1 and ϕ_i are the parameters of the i'th decentralised actor μ_i.

We do not further train $Q_i^{\boldsymbol{\pi}}$, which we have learned in Phase I, but do train $Q_i^{\boldsymbol{\mu}}$ (approximated by Q_i^v) according to:

$$\mathcal{L}(v_i) = \mathbb{E}_{\boldsymbol{o},\boldsymbol{o'}\sim\mathcal{D},\boldsymbol{a},\boldsymbol{a'}\sim\boldsymbol{\mu}}[(Q_i^v(\boldsymbol{o},\boldsymbol{a}) - y)^2] \tag{4.7}$$

where the target $y = r + \gamma Q_i^v(\boldsymbol{o'},\boldsymbol{a'})$ uses the shared task reward.

As before, the β_i parameters are interesting to examine. As we have described in the introduction, in the feudal case the critics from the first stage encode a piece of the solution to the entire problem – those aspects of the problem which do not

involve communication from worker to manager. This enables workers to quickly learn to follow the managerial commands learned in the first stage and for the manager to assign commands which do not rely on workers observations (such as information gathering commands). For commands which previously relied on access to workers' state, the managerial control will be inaccurate at first. However, as workers also optimise the overall task reward they are incentivised to communicate in such a way that they are assigned goals correctly by the manager. For the workers we set an intermediate value of $\beta_i = 0.5$. However, as we expect the managerial actions from the first stage to be (close to) ideal, given its access to the observations of all workers, we set $\beta_M = 1$.

4.3 Results

We conduct experiments in the MPE using the same hyperparameter settings as in Chapter 3, except a smaller network size for the feedforward networks for both policy and value functions (Appendix A.1.1). We show results first for the feudal case, and conclude with results for the Single-agent CPAC.

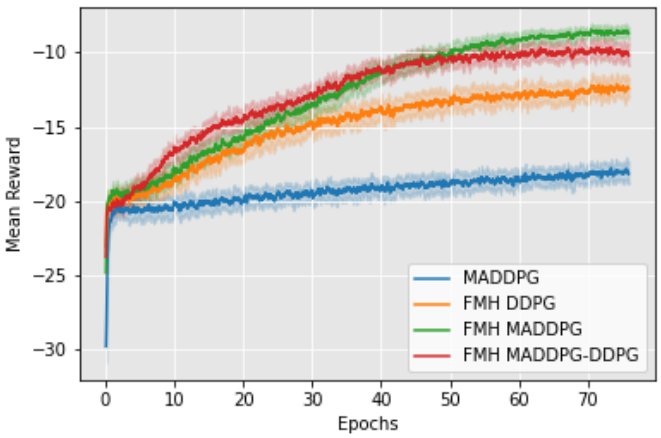

Figure 4.2: Feudal MADDPG for Cooperative Communication. We find that FMH MADDPG outperforms FMH DDPG on this problem (n=5). FMH MADDPG-DDPG appears to learn faster initially but does not reach the same asymptotic performance as FMH MADDPG.

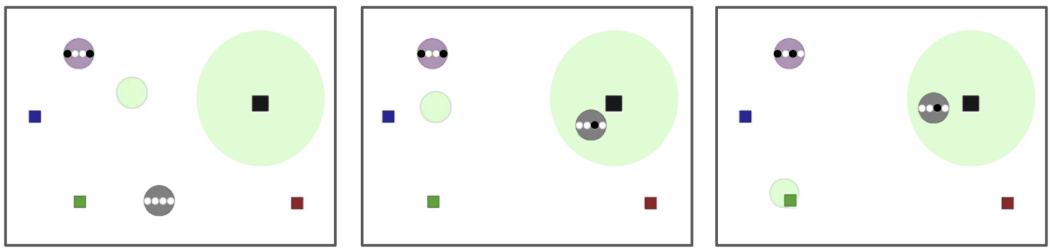

Figure 4.3: Search and Cooperative Communication. On this episode the target is the green Landmark and the green Listener must navigate towards it. The purple agent is the Speaker, which communicates with the other two agents. The grey agent is the Information Gatherer, which communicates with the Speaker. The communication space is more complex than illustrated (see Appendix A.2.3), here we overlay discrete communication messages. **Left**: The Speaker assigns the Information Gatherer to the black landmark. **Middle**: The Information Gatherer reaches the black landmark, observes the green colour of the Listener and communicates a message. **Right**: The Speaker uses this information to send the Listener to the correct green target (whilst keeping the Information Gatherer at the black landmark).

4.3.1 Feudal MADDPG

We test a version of FMH with MADDPG for each agent, as well as a mixed version with MADDPG for the manager and the remaining agents using DDPG which we refer to as FMH MADDPG-DDPG. Our task is Cooperative Communication with 6 landmarks and 5 Listeners, which had been solved by FMH DDPG with pretraining and extended communication in the previous chapter. Figure 4.2 summarises our new results; whilst FMH DDPG fails to perform well without pretraining or extended communication, FMH MADDPG is able to learn effectively. FMH MADDPG-DDPG appears to learn faster initially than FMH-MADDPG, but does not reach the same asymptotic performance. It does substantially outperform FMH-DDPG however.

4.3.2 Feudal CPAC

We wish to apply Feudal CPAC to a task in which the manager must assign a worker to gather information on its behalf and then uses the communicated information to assign another worker the correct task. To do this, we extend the 'Cooperative Communication' problem outlined in the introduction to Part I. Our new problem, called

'Search and Cooperative Communication' (Figure 4.3), also has red, blue and green landmarks, one of which is randomly selected as the target for one Listener agent at the beginning of each episode. An immobile Speaker communicates to the Listener, but now cannot see the target landmark colour, instead it receives communication from another agent called the 'Information Gatherer'. The Information Gatherer does not see the target colour either, however, there is a black landmark in the environment which will provide it with this information if it is sufficiently close to it (the large green penumbra). Solving Search and Cooperative Communication therefore requires the Information Gatherer to move to the black landmark, communicate its findings to the Speaker such that it can guide the Listener to the correct target. The overall task reward at each timestep is equal to the negative distance of the Listener from the correct target.

For FMH, we assign the Speaker the role of manager, and it therefore communicates goals to both agents simultaneously, selecting from the four possible landmarks for each. The results of our experiment can be seen in Figure 4.4. Training conventional MADDPG on this problem results in very slow learning, and performance of FMH-MADDPG does even worse, because there is no incentive for

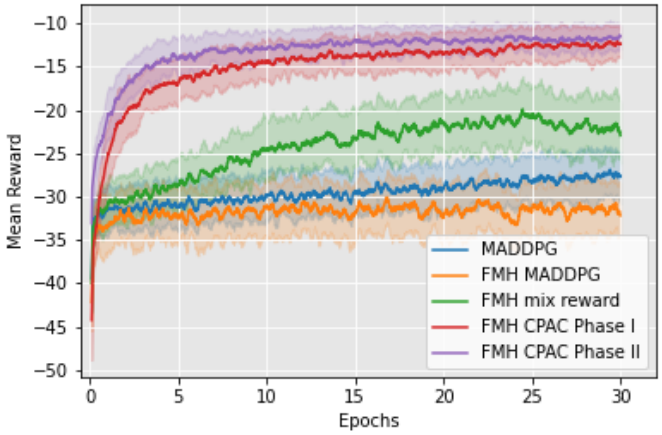

Figure 4.4: FMH CPAC on Search and Cooperative Communication. Phase I FMH CPAC training followed by Phase II elicits much better performance than alternatives (n=10).

Figure 4.5: Single-agent CPAC on Search and Cooperative Communication v2. MADDPG struggles to perform well on this task whereas our two-phase approach greatly improves performance (n=5)

the information gathering worker to communicate this to the manager. By allowing workers to receive an even mix of managerial and feudal reward, we find that performance of FMH can be improved, however, it is still far from effective. By contast, when we train the CPAC with FMH in Phase I we find that it rapidly learns to solve this task, as the manager can immediately see the gathered information. In Phase II of training, we took each model from the first phase and used it to advise training of a new model (matching each with a different random seed). We found that Phase I performance was rapidly recovered for the fully decentralised Phase II, indicating that workers additionally learned to communicate the relevant information.

4.3.3 Single-agent CPAC

The Search and Cooperative Communication problem, whilst well suited to our exploration of FMH, involves a wiring of communication which is complex, not allowing the Information Gatherer to communicate directly with the Listener. We therefore designed a new task, called Search and Cooperative Communication v2, which simplifies the communication, whilst also making the Information Gathering problem itself more difficult.

In the new task, there are 2 agents and 5 landmarks: white, black, red, blue

and green. Whereas before the black landmark always provided information when approached, now the information may be either at the white or black landmark. The Information Gatherer is provided with the colour of the target landmark and so can learn to navigate to the correct target. It now communicates directly with the Listener in order to guide it to the correct landmark (red, blue or green).

Our results for this task are presented in Figure 4.5. In Phase I we find that the single-agent CPAC is able to perform well on the task, and this performance is fully recovered in Phase II by incorporating the advice from the critic. By contrast MADDPG struggles to learn, appearing to plateau well below the performance in Phase I.

4.3.4 Conclusion

We combined MADDPG with FMH and demonstrated its effectiveness. We then extended the idea of centralisation by introducing a CPAC, which was able to effectively train agents in both the fully-cooperative and feudal settings to solve challenging information gathering tasks. This involved a two-stage training procedure and a modification to the conventional multi-agent policy gradient.

Chapter 5

Discussion and future work

We have shown how cooperative multi-agent problems can be solved efficiently by defining a hierarchy of agents. Our hierarchy was reward-based, with a manager agent which optimises the overall task objective setting goals for workers which defined their rewards. We focused initially on training using a decentralised approach but later explored centralised methods, and introduced the CPAC which helped to train information seeking behaviours in both feudal and fully cooperative systems.

In this section we describe how our approach relates to other work and outline areas for future research. We start by discussing ideas in HRL, including a preliminary experiment with continuous goals. We then review related areas in MARL and methods which combine HRL and MARL. We end by discussing centralisation, managerial shaping and the learning of hierarchies.

5.1 Hierarchical reinforcement learning

Our work drew inspiration from HRL, in particular the feudal RL architecture (FRL) of Dayan and Hinton (1993), also developed in the context of deep RL by Vezhnevets et al. (2017) in the form of Feudal Networks (FuNs). Goal-setting in FMH was achieved by specifying a relationship between the chosen managerial communication and the resulting reward function. Rather than set goals directly in the observation space of each worker, the manager utilised a 'goal space' corresponding to the possible target landmarks in the environment. Our target-based approach can be compared to multi-task formulations of single-agent RL, where it is common for

a separate goal space to be defined in addition to the immediate observation space of the agent. For example, a mechanical hand might be rewarded for rotating a cube to different orientations irrespective of the angle of each joint in the hand, with different goal orientations being selected at the beginning of each trial (this formulation is used by algorithms such as Hindsight Experience Replay (Andrychowicz et al., 2017)). In the spirit of this work, we used FMH to set goals in a subset of the full state-space, rather than micromanaging all aspects of behaviour (such as immediate velocity) present in the full observational space.

Alternative approaches from the HRL literature instead aim to generalise the setting in which goals can be set, which can be useful when a task space cannot be easily defined. For example, HIRO (Nachum et al., 2018a) sets continuous goals in the observation space of the worker, imposing structure in the form of distance from a target state. Such approaches have been effective on low dimensional tasks but typically struggle as dimensionality is scaled (Nachum et al., 2018b). Alternatively Vezhnevets et al. (2017) set directional goals in a learned hidden representation, perhaps allowing for the setting of more abstract task-relevant goals in a lower dimensional space. This is similar to the idea of 'feature control' (Jaderberg et al., 2016), which can improve performance of deep RL agents by generating auxiliary tasks which are useful for representation learning[1].

Although we did not investigate such approaches for FMH, we have conducted preliminary experiments which utilise a continuous rather than discrete communication space in which to set goals (whilst keeping movement actions in the MPE discrete). We use the Cooperative Communication problem with 6 landmarks and allow the Speaker (manager) to see the observations of its Listener (worker). The Speaker sets goals in the 14-dimensional space corresponding to the worker's full observation space (2 dimensions for velocity and 12 dimensions for relative positions of landmarks).

We try three approaches inspired by the literature:

1. The goal vector corresponds to a direction in the state space and the worker

[1] 'Pixel control', in which auxiliary subgoals are selected from the raw visual input, was also shown to be effective

is rewarded according to the cosine distance.

2. The goal vector corresponds to a target state and the worker is rewarded according to the distance from that state

3. The goal vector corresponds to a target state change for the worker which is rewarded according to its distance from the target state

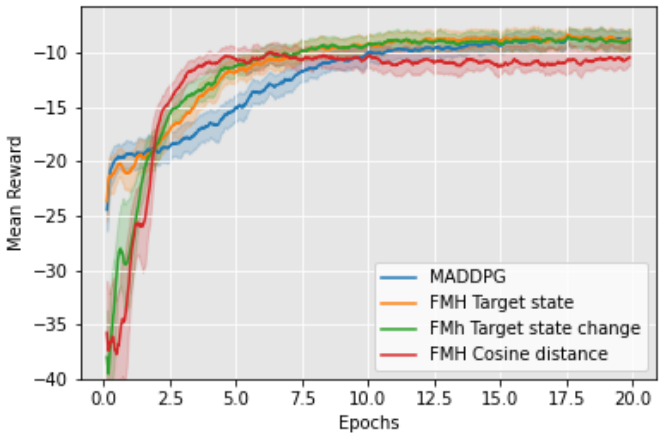

Figure 5.1: Different HRL rewards. Cooperative Communication with 6 landmarks. The manager learns to guide the worker to the correct target by specifying either the target state directly, a change in the target state relative to the worker's current state, or a cosine similarity.

We find that the manager is able to learn to control the worker using all of these methods, although approach 1 has some issues of instability such that its average performance was slightly worse (Figure 5.1). We found in practice that scaling the output of the manager by a factor of 5 improved performance substantially[2], and so we use this for all methods.

Our initial result shows promise for approaches in which the manager sets goals directly in the observational space of the worker. However, a more careful delineation between HRL methods may only be possible by applying FMH to

[2]the reason for this is unclear but is perhaps related to the saturation of the managerial output to be between -1 and 1 by a tanh function

more complex domains than the MPE. The problems we considered were generally much simpler than those in Atari from the perspective of the individual agent, and so placed less pressure on dimensionality reduction for inputs and temporal abstraction for behaviour. It would therefore be interesting to integrate the complexity of multi-agent interaction with that of single-agent RL, perhaps using recently developed MARL environments such as the Starcraft Multi-Agent Challenge (Samvelyan et al., 2019). Our multi-agent system may also benefit from being trained across multiple environments; the multi-task setting provides a powerful setting for demonstrating the benefits of hierarchical learning, as the beneficial reuse of learned skills may be more clearly demonstrated.

Our experiment in Figure 5.1 also highlights a methodological difficulty relevant to all our experiments. Whilst we optimised hyperparameters on a simpler task for the methods we considered, and generally found a single setting of these to be effective for all methods (Appendix A.1.1), it is quite possible that on more complex tasks, different settings of hyperparameters such as the learning rate or discount factor would be more suitable. Given our computational budget, extensive hyperparameter studies were not feasible, but would be important to consider in future.

We also did not investigate popular methods in HRL to increase efficiency, such as the use of hindsight and off-policy correction, and these would be expected to similarly improve learning in FMH. Interestingly, relabelling could also be attempted for generic communication which is not goal-based, an area which has hitherto seen little investigation. In a similar vein to Nachum et al. (2018a), we could ask the question; 'what message could the agent have sent to achieve the observed transition?' and relabel the message accordingly. However, unlike the goal-based case, there may be no message which satisfies this, even approximately, and so relabelling in this way may be ineffective. Instead, we have found in recent work (Ahilan and Dayan, 2020), that answering a different question is particularly beneficial; 'according to the current policies of other agents, what communication would the agent have received?'. Centralised training allows us to answer this ques-

tion and therefore to relabel messages received in the past to account for changes in the communication policies of other agents induced by learning. We find that this results in an effective 'off-environment' (Ciosek and Whiteson, 2017) correction and substantial improvements to multi-agents systems learning to communicate.

Returning to multi-agent HRL, further investigations could involve methods from unsupervised learning, such as a trajectory variational autoencoder (VAE) (Kingma and Welling, 2013; Co-Reyes et al., 2018), which can encode and decode desired state trajectories (which have been previously learned). A desired trajectory could be communicated by the manager as a setting of a latent variable, which the receiving worker seeks to replicate using its policy. Once this communication strategy has been learned, it may then be substantially easier for the multi-agent system to scale to problems involving larger numbers of agents, as we found in Chapter 3.

However, one limitation of this might be the capacity to encode sophisticated behaviour when only a single variable is communicated. An appealing idea is therefore for agents to learn to communicate sequences of variables which correspond to an agreed upon language for goal-setting. Compositional languages are highly expressive, enabling complex and abstract relationships between states and rewards to be defined. Recent work has applied this idea to address goal specification in HRL problems (Jiang et al., 2019), albeit using a language supervisor which had access to the instructions that described the scenes of the problem domain.

5.2 Multi-agent interactions

We introduced FMH to address the 'too many chiefs' inefficiency inherent to FRL, namely that each manager only has a single worker under its control. A much wider range of management possibilities and benefits arises when multiple agents operate at the same time to achieve one or more tasks (Busoniu et al., 2008). We focused on the cooperative setting (Panait and Luke, 2005); however, unlike the fully-cooperative setting, in which all agents optimise a shared reward function (Boutilier, 1996) our approach introduces a diversity of rewards, which can help with credit-assignment (Wolpert and Tumer, 2002; Chang et al., 2004) but also al-

lows for elements of competition. This competition need not always be deleterious; for example, in some cases, an effective way of optimising the task objective might be to induce competition amongst workers, as in generative adversarial networks (Goodfellow et al., 2014).

In our experiments, we introduced cases in which worker reward depended on internalised costs from collisions that workers experienced directly. A more complete range of possibilities for creating and decomposing rewards between managers and workers when the objectives of the two are not perfectly aligned, could usefully be studied under the aegis of principal-agent problems (Jensen and Meckling, 1976; Laffont and Martimort, 2009).

Our work also has connections to mechanism design (Vickrey, 1961; Hurwicz, 1973). In particular the question as to how incentives might be designed to achieve a desired objective, such as to avoid the tragedy of the commons (Seabright, 1993), in which selfish agents collectively spoil a shared resource. This idea has recently been explored in MARL (Baumann et al., 2018; Mguni et al., 2019) as a means to mitigate sequential social dilemmas (Leibo et al., 2017), and involves a manager (the 'principal') modifying the rewards received by other agents to achieve better equilibria. In contrast, we focused on explicit goal-setting by the manager to directly modulate worker behaviour.

The use of incentives other than the task incentive has also been explored from the perspective of intrinsic motivation (Chentanez et al., 2005; Baldassarre and Mirolli, 2013; Kulkarni et al., 2016). One interesting multi-agent approach is to incentivise agents to maximise their causal influence over other agents (Jaques et al., 2019). This could be added as an additional objective for a manager in FMH to learn to control its workers even whilst task performance is poor, and could potentially be used to support learning of subgoals, an approach which has achieved some success in HRL (Mohamed and Rezende, 2015; Gregor et al., 2016).

Our use of multi-agent communication also differs from a few other approaches which attempt to backpropagate through the communication channel (Sukhbaatar et al., 2016; Foerster et al., 2016; Peng et al., 2017; Mordatch and

Abbeel, 2018). This has generally improved the performance of communicating multi-agent systems and so integrating this with a feudal reward structure would be worth exploring. Finally, it would be interesting to consider how a manager might learn to allocate resources, such as money, computation or communication bandwidth to enable efficient group behaviour.

5.3 Combined approaches

Other work has also sought to combine HRL with MARL. We did not directly compare these methods to our own, generally due to their lack of an open-source implementation and due to their specificity to a particular domain. Nevertheless, we describe them here.

Zhang et al. (2009) considered a multi-level organisational structure with supervisors responsible for managing workers. They assumed that each supervisor made rational decisions and imbued them with three commands: report, suggestion and rule. A report commanded a worker to communicate its state to the manager, a rule forbade actions in certain states and a suggestion expressed preferences for worker actions in certain states. Suggestions were used to directly bias worker's exploration policies (rather than their rewards), which they found resulted in improved performance on a network routing problem. This provided an interesting example of hierarchy, although it relied on hard-coded heuristics for managerial behaviour, only using reinforcement learning for the workers.

The idea of communicating ones own subgoals was explored by Makar et al. (2001) who used the MAXQ HRL algorithm (Dietterich, 2000) to train homogenous agents which coordinated by communicating subtasks rather than primitive actions. Using hierarchy to structure communication was also explored by Kumar et al. (2017) who used a meta-controller to organise communication between agent pairs.

Application of FMH has recently been explored for multi-agent traffic management problems by Ma and Wu (2020). In this work, traffic networks were split into several regions; each region had many agents controlling the traffic signals,

which were themselves controlled by a manager agent. Each manager communicated directional goals in the form of cosine similarity, and workers were rewarded according to their achievement of these goals, as well as intrinsic reward. One difference between our approach and theirs was that both manager and workers utilised recurrent networks, whereas we only investigated feedforward policies. This was sufficient for the problems we investigated but for many partially-observed problems recurrent networks are likely to be beneficial. Overall, Ma and Wu (2020) showed that the feudal multi-agent system was able to outperform state-of-the-art alternatives according to almost all evaluation metrics used for traffic signal control.

A feudal approach has also been applied to joint order dispatching and fleet management for multi-scale ride-hailing platforms (Jin et al., 2019). Each geographical region was modeled as an agent and many such neighbouring regions were aggregated and placed under the control of a single manager. Similar to FuNs, they utilise a dilated RNN architecture and define goals using the cosine similarity. As with our approach, they use a one-to-many manager worker control scheme rather than the one-to-one approach typically found in the HRL setting.

5.4 Centralisation

In Chapter 4 we considered issues of centralisation. We began by applying MADDPG to FMH and found it to achieve better asymptotic performance than either DDPG or a MADDPG manager with DDPG workers. That FMH MADDPG learns without pretraining or extended communication suggests that centralisation can compensate for these mechanisms, which were originally introduced to address issues of non-stationarity for both manager and worker. FMH-MADDPG did, however, learn slower than the mixed FMH MADDPG-DDPG approach, perhaps due to the better scaling properties of decentralised learning. In future work we could therefore test this more carefully by experimenting with larger numbers of agents.

After applying existing methods of centralisation we then developed our own in the form of a CPAC and a two-phase training procedure. This involved training either a single-agent or multi-agent actor-critic system in a first phase of learning

on a related but simplified problem and then using the learned critic(s) to guide a multi-agent system towards solving the full problem.

Our general approach is analagous to that of a supervisor shaping the learning of a student in the all-too-familiar situation of the supervisor only having partial answers to the student's questions. Given this advice, the student is tasked with devising an optimal solution, which may be at odds with the supervisor. For example, if communication actions are costly then the first stage critics for the workers will advise against them, even whilst the optimal solution for the full problem requires the judicious use of communication. In our experiments we focused on problems where communication was not costly or limited in range, which limits this kind of interference. However, even in cases where such interference exists, the hope is that the supervisor provides advice which is useful for initial learning, and that as the student gains experience this advice can be gradually diminished. This allows the conventional convergence properties of multi-agent learning to be recovered if necessary, with the hope of finding better local optima.

When applying CPAC in the feudal case, we built on the idea that the manager in particular may benefit from centralisation. We motivated our approach as being a reverse decomposition of the 'Communication as Control' algorithm, in which the feudal system first learns control in the case where the manager can see the workers' state, followed by training the system such that workers communicate the important aspects of their observation. Whilst we found this to be effective, it is interesting to consider if this algorithm could be more directly approximated. For example, workers could benefit from an unsupervised objective to autoencode their observations, communicating the compressed representation to the manager, which is simultaneously trained to control the worker. One issue however is that workers may not learn to communicate useful aspects of their observation relevant for managerial decision making.

We also used a single-agent CPAC for the fully cooperative case, where we demonstrated its advantages in more rapidly training a MADDPG policy. One disadvantage of our approach is that it requires two stages, making direct comparison

to multi-agent training methods tricky[3]. Our approach could also be implemented in a single-stage with CPAC and MAC learning simultaneously. However, this would still not match the simplicity of conventional MAC training, as environmental roll-outs would need to be divided amongst the MAC and CPAC policies.

The idea of using a centralised policy expert to guide the learning of a multi-agent system has been explored recently by Lin et al. (2019) who begin by treating it as a supervised learning problem in which each agent seeks to imitate expert behaviour. However, they also acknowledge the problem of learning communication in this scenario and therefore they introduce a separate communication loss, which they optimise using backpropagation. Using centralised expertise has also been explored by Chen (2019), who aim to distil the single-agent expert policy into multi-agent network policies, and Lee and Lee (2019) who mix demonstrations from a centralised policy to aid multi-agent training. It would therefore be beneficial to compare these ideas to our own approach of altering the multi-agent policy gradient, to discover what the relative advantages and disadvantages are.

It would also be worth seeing if our method can be used to address more complex problems. In particular, the single-agent CPAC may struggle to scale to large action spaces, and it may be possible to resolve these issues by using attention-based mechanisms (Vaswani et al., 2017; Iqbal and Sha, 2019) or imposing structure on the outputs of the neural network policy tailored to the natural divisions of the multi-agent action-space.

We also only applied our method to fully cooperative problems and it may be possible to apply these ideas in the competitive setting by equipping each agent with a CPAC. If an agent can see its opponent state, it may aid it in learning effective response strategies, albeit care must be taken to avoid forms of cheating. A related idea has in fact been studied by Vezhnevets et al. (2019), who were interested in opponent interactions of competing agents with concealed information. One problem they considered was a spatialised version of rock/paper/scissors, where opposing agents navigate an environment to pick up their weapon of choice and then can

[3]for example, whilst many episodes may be used for single-agent CPAC training, wall time may be considerably shorter as only a single agent need be updated

encounter each other, recieving rewards according to the match-up of their chosen weapons. Vezhnevets et al. (2019) general approach involved exploiting centralised learning by factorising the value function using the concealed information of the opponent (this might include, for example, that the opponent was carrying a rock). This latent space was then reused to train a factorised decentralised policy for the agent. This mixture policy approximated inference of the latent solely from its history of local observations and used this to weight a behavioural response.

Finally, when training the centralised critics in all cases we simply used the aggregated observations and actions of all agents. For Search and Cooperative Communication this is insufficient to make the environment Markov, and so it would be worth considering if it would be useful to augment the critic with the hidden environment information. In fact, this idea is not specific to multi-agent learning, being applicable even in the single-agent partially-observable case. Perhaps surprisingly, we are not aware of explicit investigations into the application of this idea, and so evaluating its effectiveness may be worthwhile.

5.5 Shaping

The question of how we might help RL agents to learn has been richly studied in the literature and is relevant to our work. Reward shaping involves altering the rewards in the environment to facilitate the learning of agents solving a particular task (Ng et al., 1999). It is therefore interesting to consider if the manager could learn to do so for its workers, for example by determining the shape of the reward function.

We only considered rewards propotional to the negative distance from the target state, which likely aided learning far more than simply specifying a narrow region around the target in which the worker receives reward. However, a more detailed investigation of different shaping functions would be interesting, including whether a manager might be able to specify a metric along with a target state which would weight the distance of the worker from the goal state according to each dimension's importance, as determined by the manager.

The manager may also guide learning by ordering the tasks presented by the

manager to the worker. This idea of starting with easy problems and gradually increasing their difficulty is known as curriculum learning (Skinner, 1990; Krueger and Dayan, 2009; Bengio et al., 2009). An example of such an approach for RL was explored by Florensa et al. (2017) who trained a robot in reverse to gradually reach a goal state from a set of target states increasingly far from the goal. These start states were generated adaptively according to the performance of the agent.

5.6 Learning hierarchies

The hierarchies used in FMH were simple in structure and specified in advance, based on our knowledge of the various information and action capabilities of the agents. It would be interesting to develop mechanisms for the formation of complex hierarchical structures.

We may represent the managerial hierarchy as an adjacency matrix with non-zero entries indicating that the agent in row i manages worker in column j. For example, if agent 1 manages agent 2, and agent 2 manages agent 3, then the representation would be:

$$D = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \tag{5.1}$$

We would like hierarchies to correspond to directed acyclic graphs (DAGs), and this correspondence may be verified by checking that D is nilpotent; there exists some positive integer n such that $D^m = 0, \forall m \geq n$ (in this particular three-layer hierarchy $D^3 = 0$). If the column j is empty it indicates that agent j is unmanaged and so optimises the task reward. Similarly, if row i is empty it indicates that that agent is the lowest agent in the hierarchy without any management responsibility.

In this thesis we employed hierarchies with one manager and all other agents as workers. This would therefore be represented as:

$$D = \begin{pmatrix} 0 & 1 & \dots & 1 \\ \vdots & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \end{pmatrix} \tag{5.2}$$

where agent 1 is the manager. An extension to this representation would allow values in the matrix to range between 0 and 1, expressing the degree of control the manager has over rewards, with the remainder corresponding to task or intrinsic rewards.

In general, more complex hierarchies may enable more effective learning and performance and so we would like a method to find them. One approach might be to start simple and steadily increment the complexity of the managerial hierarchy over time. A 0 matrix could be intialised, which defines a shared reward flat multi-agent system, and then modified as an outer loop of hyperparameter optimisation. Modifications could correspond to randomly changing individual elements of the matrix from 0 to 1 (whilst ensuring the DAG structure is maintained) and continuing training to see if performance improves.

Ideally many models would be trained simultaneously, with more promising hierarchies being replicated, as in population based training (Jaderberg et al., 2017). Population based training allows for hyperparameters to continuously be adapted throughout the training process, potentially enabling the complexity of the hierarchy to be gradually grown by building off previous complexity. As an approach to learning hierarchies, this is reminiscent of Neurath's ship, who opined that 'we are like sailors who on the open sea must reconstruct their ship but are never able to start afresh from the bottom'.

It would also be interesting to see if more complex methods for modifying the hierarchy might be useful. For example, a useful prior might be to bias against assigning any worker more than one manager such that there is no conflict of control. Additionally, a preponderence towards assigning existing managers new worker agents, rather than creating more managers might be useful. This 'rich-get-richer' property is reminiscent of the Chinese Restaurant Process in Bayesian

non-parameterics (Griffiths et al., 2004), as well as society at large.

Finally, it is worth considering if we can say anything about the optimality of hierarchies for a given problem (as was done by Solway et al. (2014) in the single-agent case). According to our 'Communication as control' algorithm, we might be tempted to argue that the hierarchy we have primarily examined with only a single manager and many workers is ideal because we can treat it as a direct approximation to single-agent reinforcement learning. However, as we have noted, decentralisation has the potential to scale much better than centralised approaches. It would seem therefore that for a given problem it may be the case that some agents would benefit from hierarchical control whereas others might benefit from being left to their own devices. This intermediate approach might best trade-off the advantages and disadvantages of single-agent and multi-agent learning. For agents which do end up managing, a further question arises as to who manages the managers[4]. An intuition for adding layers to the hierarchy would be increasing benefits of temporal abstraction, as with HRL, as well as allowing for better global decision making from managers receiving communication from a larger number of workers. However, we are not currently able to provide a precise presciption of when managers should exist and what exactly they should be doing. Fortunately, this does not stop us from promoting them.

[4] we can say that top-level managers are in effect managed by the task

Part II

World Models

It was just a story about people and rats. And the difficult part of it was deciding who the people were, and who were the rats.

— Terry Pratchett

Chapter 6

Introduction

6.1 Adapting to a structured world

Humans and other animals inhabit a world replete with statistical structure and regularities over many spatial and temporal scales. Such structures may arise for a multitude of reasons. One source is the simplicity and generality of physical laws, which give rise to regular phenomena such as the daily rising and setting of the sun, or the formation of stalagmites and stalactites within the confines of caves. Another more complex source of structure is that which is imposed by other humans or animals, such as the regular timetables of workers in a company or the complex maze in which a laboratory rat is placed.

In cases where structure influences an animal's ability to achieve reward, animals are often able to make decisions which exploit these regularities, for instance by reacting faster to more probable events (Niemi and Näätänen, 1981). The mechanisms which enable animals to make effective decisions is a key question in cognitive computational neuroscience (Dayan and Daw, 2008). It has long been established that there are multiple such mechanisms in the brain. This multiplicity not only provides redundancy, but also provides various computational and statistical advantages.

Perhaps the simplest way which enables animals to respond effectively to structures in their environment is through evolutionary 'hard-wiring' or non-adaptive instinctive behaviours (Breland and Breland, 1961). These can gradually emerge

through evolution if regularities are consistently experienced across many generations of a species. For example, the consistent rising and setting of the sun has lead to the development of sleep cycles according to circadian rhythms. Similarly, in the case of food, Pavlovian approach behaviours are almost always beneficial and so are hard-wired into some animals as an unconditioned response. This was elegantly demonstrated by Hershberger, who designed an experiment in which hungry chicks approaching a chicken-feeder (which had previously learned this was a source of food) would find the chicken-feeder moving away twice as fast, whereas moving away from the chicken-feeder would lead to it moving towards the chick twice as fast (Hershberger, 1986). Hershberger showed that the chicks, even with training, could not learn the correct behaviour of moving away from the chicken-feeder in order to receive food. This 'Pavlovian controller' (Dayan et al., 2006), is particularly illuminating as it highlights systems in the brain which are highly inflexible yet nevertheless give rise to appropriate behaviours in the vast majority of situations, due to the consistency of certain aspects of the environment across many generations. Furthermore, these methods are both statistically and computationally cheap, neither having to be learned during an animal's lifespan nor constantly recalculated each time relevant situations occur.

When the structures which influence an animal's experience are different from those experienced during its evolutionary history, inflexible control mechanisms are insufficient. Nevertheless, in many cases animals often learn to adapt effectively, indicating the existence of further mechanisms for prediction and control which learn directly from experience, as in reinforcement learning. One such prediction mechanism shares close connections to the theory of model-free RL, with evidence for its instantiation in the dopamine system of the brain (Schultz et al., 1997). The release of dopamine, according to this model, corresponds to the reward prediction error signal of temporal difference learning, and enables state and action-values to be learned even when transitions between states are not learned. This process is in practice often statistically inefficient, as values typically require a large amount of experience to be learned, but computationally efficient, as cached values can be

used directly to select actions. Control using the model-free system is therefore quick to use and provides a natural model for habitual behaviours.

An alternative system, connected to the theory of model-based reinforcement learning, uses learned transitions between states to plan outcomes and take actions accordingly. In contrast to the model-free system, this is often statistically efficient (if the underlying transition structure is easy to learn) but computationally inefficient, as when used for planning the number of possible outcomes grows exponentially with depth. The model-based system is typically flexible, and can be used to rapidly devise new stategies upon changes to the world (such as the blocking of a passage in a maze) or changes in the reward structure.

Both model-free and model-based systems occupy different sweet spots in the trade-off between statistical and computational efficiency. Animals may arbitrate between them optimally (Daw et al., 2005), which often results in a transfer from more model-based strategies early in training to model-free strategies when greater experience has been gained[1].

Model-based learning involves the animal understanding the transition structure between states of the world to enable effective planning. The critical question of what consititutes 'state' however, rapidly emerges; planning in the 'raw' observational space is unlikely to be useful, whereas planning in a more abstract representational space which reflects the structure of the task is potentially far more useful. The notion that animals build such 'world models' (Daw et al., 2005, 2006; Gläscher et al., 2010) has its roots in the idea of cognitive maps (Tolman, 1948; Behrens et al., 2018) as first introduced by Tolman. A cognitive map is in general defined as an internal, mental model of the world which an animal uses to guide its behaviour (through inference or planning). An important piece of evidence in favour of cognitive maps was Tolman's discovery of latent learning, in which rats learn the structure of a maze even in the absence of reward, and can adaptively use this information when rewards are later introduced. For example, rats could learn to take shortcuts to reach rewards and find new routes when old routes were blocked

[1] it is also worth noting the possibility of an episodic controller which may be particularly important in the regime where experience is extremely limited (Lengyel and Dayan, 2008)

(Tolman et al., 1946). This behaviour could not be straightforwardly accounted for by models which describe behaviour in terms of simple stimulus-response mappings in pursuit of reward.

The neuroscientific evidence for a cognitive map was further supported in the spatial domain by the discovery of place cells in the hippocampus, which provide a population code for representing space (O'keefe and Nadel, 1978). Later studies uncovered a plethora of cell types, including grid-cells in the medial entorhinal cortex which fire at multiple place fields on a triangular grid as well as cell types which encode head direction, relationships to borders and even to the locations of other agents. Such abstract representations are highly divorced from the immediate sensory perception of the animal, yet are of enormous functional relevance to a variety of spatial tasks which the animal might face.

The exciting discovery of abstract neural representations in the spatial domain spurred the natural question as to whether such neural machinery is also used in non-spatial domains to code for structural knowledge. Recent research has provided strong support for this hypothesis; for example, in a task in which rats must use a joytstick to manipulate sound along a continuous frequency axis, cells types known to be place or grid cells in the spatial domain will instead represent frequency in an analagous way (Aronov et al., 2017). Similarly, humans navigating conceptual two-dimensional knowledge showed a grid-like signal in regions also activated during spatial navigation (Constantinescu et al., 2016).

Evidence for representations of abstract structures also extends beyond the hippocampus and entorhinal cortex. In his work on 'learning sets', Harlow trained monkeys to discriminate between pairs of objects in order to receive reward (Harlow, 1949). However, these pairs would only appear for a small number of trials (typically 6), known as a 'set', before a new pair of items was chosen for the next set. The task followed a simple rule; the rewarded item on a particular set would remain the rewarded item for the rest of the set. Thus, whilst on its first choice the monkey only had 50 percent chance of picking the correct item, perfect performance was possible on subsequent trials by following a win $\rightarrow$ stay, lose $\rightarrow$ shift

strategy. Harlow found that monkeys were able to learn to do this effectively, even in cases where stimuli which were rewarding on a previous set became unrewarding on the current set (or vice versa). This showed the ability of monkeys to go beyond discrimination learning and form a basic understanding of the task. Interestingly, lesions to ventral prefrontal cortex (vPFC) (orbitofrontal and ventrolateral prefrontal cortex) in macaque monkeys abolishes this ability (Walton et al., 2010; Rudebeck and Murray, 2011). The behaviour of animals with such lesions are predicted by a reinforcement learning agent which only learns from immediate sensory observations rather than assigning credit to abstract states (Wilson et al., 2014). Taken together, these various streams of research have shown the ability of humans and animals to form task-relevant world models, with connections to a number of brain regions.

6.2 World models and partial observability

We investigate the capability of rats to use a world model in a reward-based task. For such world models to be useful they must not only be accurate but also easy to use. A critical aspect of this is being able to infer the appropriate hidden states (or latent variables) given imperfect sensory observations. In general, and for the task we investigate in this work, the sensory evidence at a given moment often provides insufficient evidence as to the hidden state, a problem known as partial observability. To achieve accurate inference, an animal must remember and correctly integrate evidence provided by past observations, which demands the effective and adaptive use of forms of working memory (Zilli and Hasselmo, 2008; O'Reilly and Frank, 2006; Todd et al., 2009).

An appropriate formalism for decision making in the partially observable case, known as the POMDP, was introduced in Section 1.4.3. This formalism, when combined with a correct world model of the underlying hidden process and with the forward-backward message passing algorithm, enables computation of a belief state over likely values of the hidden state which incorporates all relevant history (Kaelbling et al., 1998). Whilst the POMDP formalism allows for actions taken by

the agent to affect future states, we focus here on a prediction task in which actions do not affect state, and for which an HMM (Section 1.4.2) provides a more natural model.

The latent variables relevant to a task may persist over a variety of timescales. For short timescales of a few seconds, research has often focused on accumulate-to-bound decision making (Ratcliff and Rouder, 1998; Gold and Shadlen, 2002) or persistent activity states (Miyashita, 1988; Fuster, 1997; Frank et al., 2001). Very long times, perhaps even across days, are associated with macro-states or contexts (Haruno et al., 2001; Gershman et al., 2010). By contrast we consider a task in which the critical structure (which supervenes over shorter-time task requirements) typically concerns an intermediate scale of tens of seconds.

In the following chapter we investigate aspects of the behavioural data of rats engaged in a task with hidden structure. As we will see, rats learn to use this medium-term structure to predict oncoming states and adjust their actions accordingly. However, their behaviour reveals errors which, when they arise, result from chance recent observations that are misleading as to the identity of the hidden state. We show how to account for their performance by building an HMM which characterises the environment, and in which evidence from past observations is imperfectly integrated with recent observations. We find that as training progressed for the subjects, they better learned to predict oncoming states. This reveals to us a process by which subjects learn to use past evidence more effectively to infer their state in the world.

Chapter 7

Experiments and model

The results of this chapter were presented in:
PLoS computational biology (Ahilan et al., 2019)

7.1 Task and experiment

We consider a cumulative handling time task (Breton et al., 2009; Solomon et al., 2017) in which rats hold down a lever for an experimenter-defined time period, called the price (P), in return for rewarding electrical stimulation of the medial forebrain bundle (Olds and Milner, 1954) at a fixed current and a given pulse frequency (f). In this paradigm, subjects experience many trials, each of which consists of an epoch during which price and frequency are fixed. Subjects may achieve the price cumulatively, over multiple presses during the trial. The duration of a trial (D) is 25 times the price (except for a minority of trials with price less than 1 second which last 25 seconds) allowing for many rewards to be obtained. This duration excludes a short, typically two second period following each reward termed the 'black-out delay' which allows for reward consumption and during which the lever is first retracted and then re-extended. Together frequency, price and duration define the experimentally-set parameters of a given trial. As these parameters vary, subjects face trials with different costs and benefits; previous studies have used their resulting responses to understand the subjective tradeoff between labour and leisure (Breton et al., 2013; Niyogi et al., 2014a; Solomon et al., 2017). Along with those authors, we focus only on variations in frequency and price and not duration, as the

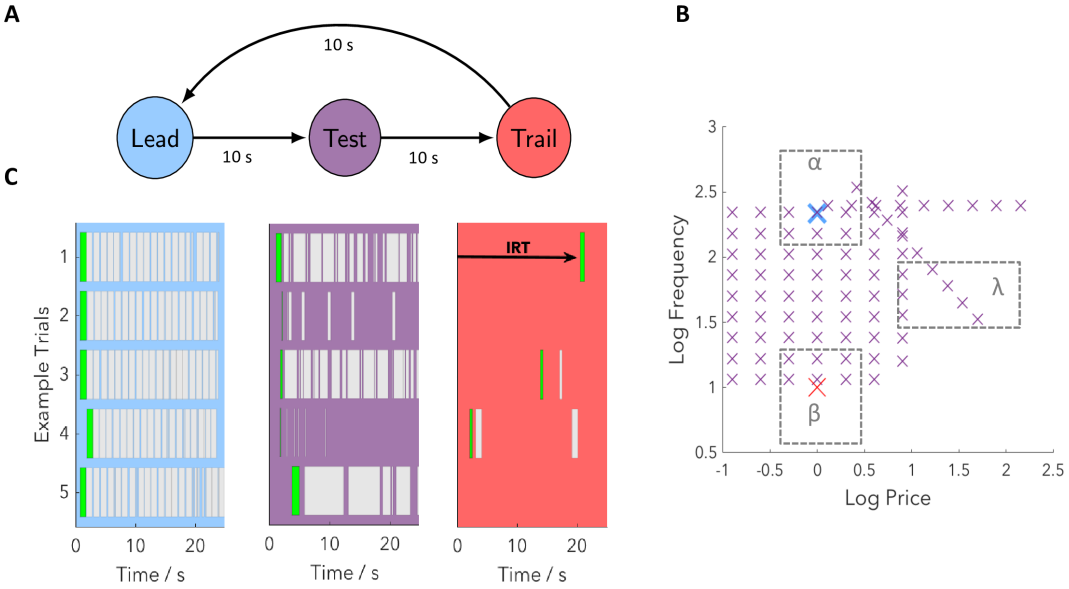

Figure 7.1: The structure of the experiment. (**A**) Trials come in a predictable cyclic triad. Each trial corresponds to a period of time where price and frequency are fixed. The intertrial interval is 10s. (**B**) Frequencies and prices associated with each trial type (subject 1). Lead trials are highly rewarding with a fixed high pulse frequency and a short price (blue cross). Trail trials are negligibly rewarding with a fixed low pulse frequency and a short price (red cross). Test trails vary in frequency and price from trial to trial and so are variably rewarding (purple crosses). In addition to the crosses we also define regions α, β and λ (dashed grey rectangles) which are relevant for Fig 7.4. Note that in regions α and β, test trials are similar to lead and trail trials respectively, whereas in region λ test trials are dissimilar to both. (**C**) Responses from five example triads of trials from a trained subject (subject 1). Grey bars correspond to the lever being depressed, with initial responses highlighted in green. Pressing is almost continuous on lead trials, varies on test trials from trial to trial (only the first 25s is shown) and is rare on trail trials. We label the IRT, which reflects subjects' beliefs about the rewarding nature of the current trial before they have experienced any within-trial evidence.

latter depends directly on price, being (almost always) directly proportional to it.

At the beginning of each trial, a high frequency stimulation train, called a prime, is delivered. The subjects are then free to choose whether and when to engage with the lever. We analyse two major dependent variables. The first is the engagement probability (EP), which is the probability that subjects engage with the lever at all. The second, if subjects do indeed engage, is the initial response time (IRT), which is the time it takes them to first press the lever following the prime; we define these in more detail in Appendices B.1.

Trials come in a predictable cyclic triad consisting of 'lead', 'test' and 'trail' trial types (Fig 7.1 A) separated by a fixed intertrial interval of 10s. Each trial type is associated with different frequencies, prices and durations (Fig 7.1B; shown with log base 10 here and subsequently), but are otherwise identical. When subjects know the frequency and price associated with a trial, and hence the worth of work, they typically choose an appropriate level of engagement with the lever. This is illustrated by the ethograms in Fig 7.1C in which the lever presses of a trained subject are plotted for different trial types (we ignore the post-reward 'black-out delay').

For lead trials, which correspond to fixed, high-frequency stimulation with a short price of 1 second, subjects typically work the entire duration of the trial, as the high-frequency stimulation is highly rewarding. By contrast for trail trials, which have fixed, low-frequency stimulation at the same short price of 1 second, subjects barely work. Test trials, which involve a range of frequencies and prices which change from trial to trial (but are fixed across a particular trial) give rise to variable amounts of work, depending on the particular values of the frequency and price.

The data in the present paper are drawn from Solomon et al. (2017) which describes in detail all aspects of the experiment, including training prior to the full task. Training involved a shaping protocol which eventually introduced lead, test and trail trials, enabling subjects to learn the cyclic triad structure. It used a more limited range of test frequencies and prices than was ultimately employed in the main experiment (Figure 7.2).

We studied a total of six subjects, each of which had experienced approximately 1500 triads of trials over a period of weeks. To allow adjustment from training to the full task we excluded the first 126 triads from our analysis, corresponding to one complete survey of the test trial frequencies and prices as defined in Solomon et al. (2017). The number of surveys analysed for subjects 1-6 was 12, 10, 11, 8, 12, and 13 respectively, with each survey being acquired over 2 daily sessions, lasting approximately 6 to 7 hours each. Subjects 1-6 in this paper correspond to subjects F03, F09, F12, F16, F17 and F18 respectively in Solomon et al. (2017). Whilst

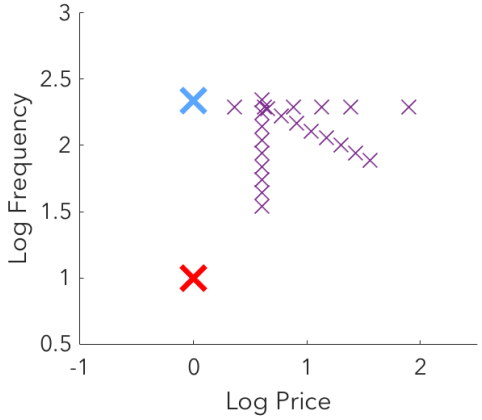

Figure 7.2: Frequency-price. Training with the triad structure involved a more limited range of frequencies and prices (subject 1)

in general, the results we describe apply to all six subjects, for simplicity we often display results in full for only subject 1, describing the remaining subjects using summary statistics. We report significance for individual subjects at the $P < 0.05$ level, with further details on exact p-values and of our methodology being provided in Appendices B.2 and B.3.

7.2 Results and model

7.2.1 Subjects learn the task transition structure

Previous analysis of these data has primarily focused on behaviour during test trials, and in particular on responses occurring after the initial responses (Niyogi et al., 2014a,b; Solomon et al., 2015, 2017). Following Breton (2013), we instead considered all trial types, and primarily focused on initial responses, since they reflect the subjects' beliefs about the likely worth of a trial before they encounter any within-trial information. They are thus the best source of information about the subjects' understanding of the cyclic triad structure. We characterised the initial responses by EPs and IRTs.

Fig 7.3 contrasts the performance of subjects when they have just begun training in the triad structure with the performance of the same subjects after they have been trained. For this analysis we exclude the first 5 triads during training as subjects were not always engaged in the new task when it first began but quickly learned

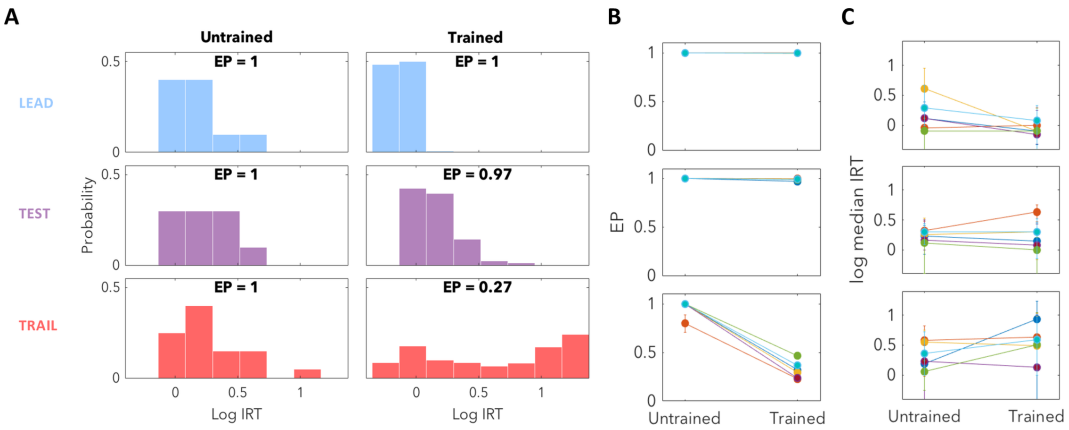

Figure 7.3: Subjects learn to predict oncoming trials. (**A**) We compare the responses of subject 1 when it has just begun training with the triad structure to its responses once trained. Early in training (left) the subject responds with short IRTs for all three trial types and EPs of 1, reflecting engagement in the task but an inability to predict the oncoming trial type. After training (right), IRTs reflect accurate prediction of oncoming lead and test trials, with certain engagement and rapid but generally distinguishable responses on the two trial types. For negligibly rewarding trail trials, the subject responds appropriately in the majority of cases, as indicated by both a low EP and a number of responses with long IRTs. However, in a minority of cases subjects also responded with short IRTs, which indicates inaccurate prediction of the trail trial. (**B**) For lead and test trials, EPs remained close to 1 (subject 1's response in dark blue). On trail trials, EPs were found to decrease consistently for all 6 subjects (binomial proportion test; h_3). (**C**) For lead trials, median IRTs remained short, and for 4/6 subjects became even shorter once trained (permutation test; h_4), as subjects learned to predict the highly rewarding lead trial. For test trials, with their lower expected rewards, median IRTs remained relatively constant and were longer in trained subjects than lead trial IRTs for all subjects (permutation test; h_2). For the poorly rewarding trail trials, median IRTs appeared not to change consistently, but we examine the properties of the trail trial distribution in more detail in Fig 7.4.

to be. Analysis of the subsequent 20 triads for each subject revealed this, with EPs close to 1 and short IRTs for all trial types. These rapid and reliable responses likely reflected the subjects' lack of understanding of the task structure, as if they predicted engagement with the lever to be valuable, or at least worth exploring, on all trial types. This was further supported by the finding that 5/6 subjects did not respond with a median trail trial IRT which was significantly longer than the median IRT of a combined distribution of lead and test trials (permutation test; h_1). For the significant subject, the median trail trial IRT was not large (3.15s) and the EP was 1, and so this likely reflected initial stages of learning.

After the training period, the same subjects emitted very different initial responses for the different trial types. To a first approximation, the difference in these initial responses for trained subjects reflected the expected worth of the trial: the larger this worth, the greater the EP (up to a maximum of 1) and the shorter the IRT.

For lead and test trials, the EP was generally very close to 1, with test trial IRTs being slightly longer for all subjects than those for (the on average more valuable) lead trials (permutation test; h_2). That test trial IRTs were longer than lead trial IRTs is interesting as this behaviour is seemingly suboptimal – subjects need to explore to find out the test trial's value before they can determine the appropriate response, and waiting at the beginning of a trial reduces their potential to exploit the test trial if it is indeed of high value. We therefore interpret the longer latency on test trials as indicating a sub-optimal Pavlovian response to an accurate prediction of relatively lower expected future reward, an effect which has been observed elsewhere (Liu et al., 2000; Dayan et al., 2006). This type of response is convenient for our purposes as it indicates that subjects learned to accurately and differentially predict lead and test trials, even before they engaged with the lever.

Subjects responded very differently on the negligibly rewarding trail trials. EPs were typically small, and when subjects did engage, the resulting IRTs were often long. However, on a substantial fraction of occasions, the IRT s were instead short, which is surprising because trail trials were designed to be effectively worthless to the subject. We explore the possibility that the pattern of long and short IRTs is a signature of subjects' inability to predict trail trials perfectly, and are thus a result of erroneous inference. They therefore provide a window into the subjects' inferential processes.

7.2.2 Misleading evidence leads to mistaken state inference

Trail trials are preceded by test trials, which involve a range of different frequencies and prices. Some of these conditions resemble either lead (region α, Fig 7.1B) or trail (region β, Fig 7.1B) trials. According to the task transition structure, lead or trail trials are followed by test or lead trials respectively, both of which are associated with high EPs and low IRTs in subjects' initial responses. We therefore

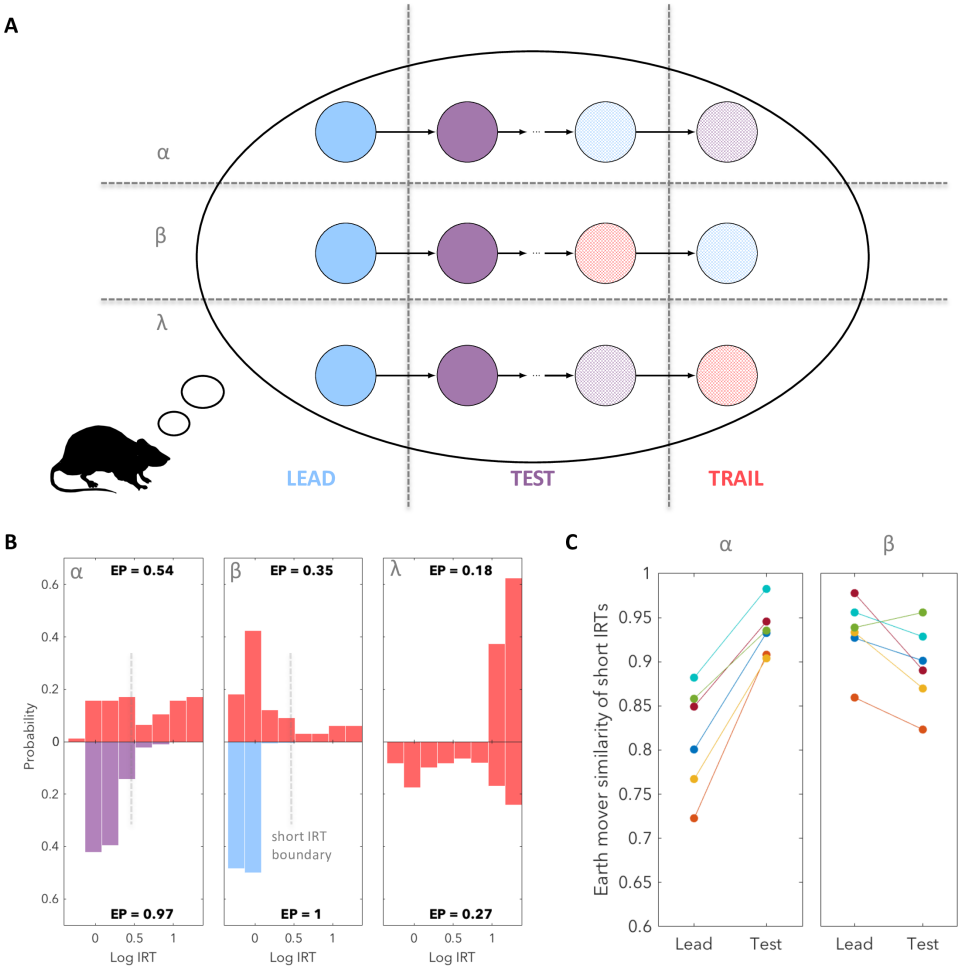

Figure 7.4: Short IRTs on comparatively worthless trail trials as mistaken inferences. (**A**) When a test trial had a frequency and price similar to that of either a lead trial or a trail trial (regions α and β respectively; Fig 7.1B), the subject's belief about the trial type may have been mistaken. If this incorrect belief was combined with a correct understanding of the transition structure of the task, the subject would have expected the next trial to be a test or lead trial respectively, rather than a trail trial, and so would have chosen a short IRT rather than no response or a long IRT. By contrast, if the frequency and price were dissimilar to both lead and test trials (region λ; Fig 7.1B), the test trial would be unambiguous and the subject would either not respond at all, or would elect a long IRT on the predicted trail trial. These effects are probabilistic, which we indicate by lighter shading. (**B**) We tested this hypothesis by examining trail trial responses given that the preceding test trial's frequency and price were in regions α, β or λ. Mirror plot histograms, subject 1. Left: When test trials had similar frequency and price to lead trials (region α), the resulting distribution of short trail trial IRTs (upper) was test-like (the lower left plot shows the actual distribution of IRTs on test trials). Middle: when test trials were similar to trail trials (region β) the resulting distribution of short trail trial IRTs (upper) was lead-like (lower plot). Right: when test trials were dissimilar to both lead and test trials (region λ), short IRTs were no longer observed (upper) despite these responses being common in the trail trial distribution over all preceding frequencies and prices (lower) . (**C**) This confusion effect is trial type specific. Short IRTs on trail trials following test trials in region α are more similar to test trial IRTs than to lead trial IRTs for all 6 subjects (permutation test; h_5). Similarly, short IRTs on trail trials following test trials in region β are more similar to lead trial IRTs than to test trial IRTs for 3/6 subjects with the difference not being significant for the remaining subjects (permutation test; h_6).

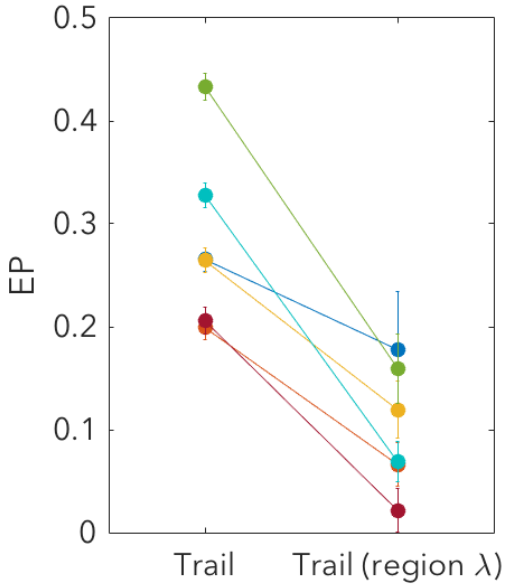

Figure 7.5: Filtering reduces EP for trail trials. When trail trial responses are filtered such that only those with preceding test trials in region λ are included, a decrease in the EP is observed

considered the possibility that short IRTs on trail trials arose when the subjects had been confused by the preceding test trial, but had applied their good knowledge of the transition structure (Fig 7.4A; see also (Breton, 2013)).

To test this hypothesis, we sorted the trail trial IRTs by the frequency and price of the previous test trial (Fig 7.4B). Indeed, when the test trial had similar frequency and price to a lead trial (region α), the resulting distribution of short trail trial IRTs resembled that of a test trial. This is consistent with the subject inferring the test trial to be a lead trial and hence the subsequent trail trial to be a test trial. Likewise, we found that when the test trial was similar to a trail trial (region β) the resulting distribution of trail trial IRTs was similar to that of a lead trial, again consistent with expectations. For test trial frequency-price combinations dissimilar to those of either lead or trail trials (region λ, Fig 7.1B), subjects were rarely confused, and so short IRTs occurred much more rarely and EPs were much lower (Figure 7.5).

To quantify whether the short IRTs sorted in this way are more lead-like or test-like respectively we calculated the earth mover's similarity between these distributions and lead and test distributions (Fig 7.4C). We define the earth mover's

similarity to be 1 minus the earth mover's distance (or equivalently 1 minus the Wasserstein distance). To select only short IRTs, we eliminated IRTs greater than the 95th percentile of the test trial distribution. We found that for all 6 subjects, responses on a trail trial following a lead-like test trial were significantly more test-like than lead like (permutation test; h_5). Similarly, responses on a trail trial following a trail-like test trial were significantly more lead-like than test-like for 3/6 subjects (permutation test; h_6).

Having discovered this confusion effect, we investigated it in more detail by considering the separate influences of frequency, price and duration. We found that frequency strongly influenced subjects' inferences (Fig 7.6 A): for intermediate, and therefore not misleading, frequencies, subjects were much less likely to respond rapidly on a trail trial, even when price and duration were misleading. Similarly, we found subjects were sensitive to price and/or duration (Fig 7.6B), as when these were long, and therefore not misleading (since lead and trail trials ubiquitously had price of 1s), subjects were much less likely to respond rapidly even when the frequency was misleading. Finally, we also examined the minority of cases when price was not misleading but duration was (Fig 7.6C). These arose when price was less than 1 second, as the duration was fixed at 25 seconds rather than being 25 times the price, as otherwise. We show that subjects were sensitive to the price on the preceding test trial when its frequency was high but not when it was low. We speculate about the reason for this in the discussion, but do not seek to model it as its effect is subtle, only influencing a fraction of the data.

Whilst the description of confusion outlined in Fig 7.4 provides a clear, model-agnostic, account of the varied responses on trail trials, it only provides a simplified, deterministic picture of this process. We therefore built a probabilistic model, incorporating our understanding from Fig 7.6A;B, in order to describe this more precisely.

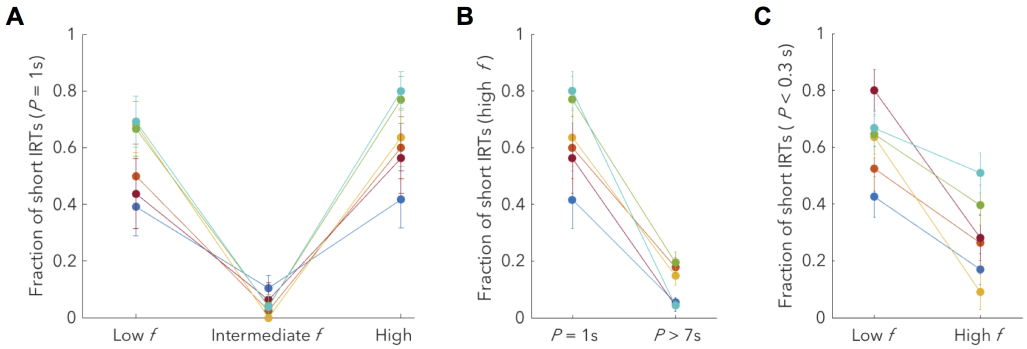

Figure 7.6: Subjects use multiple sources of evidence from the preceding test trial to determine a response on the trail trial. (**A**) Intermediate test trial frequencies only very rarely lead to short trail trial IRTs, even when price and duration are misleading. This indicates that subjects can use frequency to determine the appropriate response when this frequency is different from that of lead or trail trials; see materials and methods for a definition of these regions. This difference is significant for all subjects when comparing intermediate to both 'Low f' (binomial proportion test; h_7) and 'High f' (binomial proportion test; h_8) categories. (**B**) High test trial prices also only rarely lead to short trail trial IRTs even when frequency is misleading (here we show for high, lead-like frequency) . As duration is perfectly correlated with price for prices of 1 second or greater, this indicates that subjects can use price and/or duration to determine the appropriate response. The difference between the two categories is significant for all subjects (binomial proportion test; h_9). (**C**) When test trial price is short, test trial duration remains at 25 seconds, thus we consider cases in which price is not misleading (< 0.3s) but duration and frequency are. Short trail trial IRTs depend on the frequency of the preceding test trial. When the frequency is low, subjects respond with a similar fraction of short responses as for a price of 1s (Figure 4A; left), indicating price insensitivity. However, when the frequency is high, subjects are price sensitive, with a decreased fraction of short responses. This difference was significant for 4/6 subjects (binomial proportion test; h_{10}).

7.2.3 Modelling the inference process

The task itself can be described in the form of an HMM, with hidden states representing the trial type, and a binary transition matrix reflecting the deterministic cyclic triad structure. Given the predominant regularities in responses, highlighted earlier, we assume that subjects have learned this essential structure, associated with a task transition matrix (**A**), which acts on the subject's belief state when transitioning between trials.

Fig 7.7 captures the key steps of correct and approximate inference. First, we assume that at the end of a lead trial, the subject is correctly certain that this is its

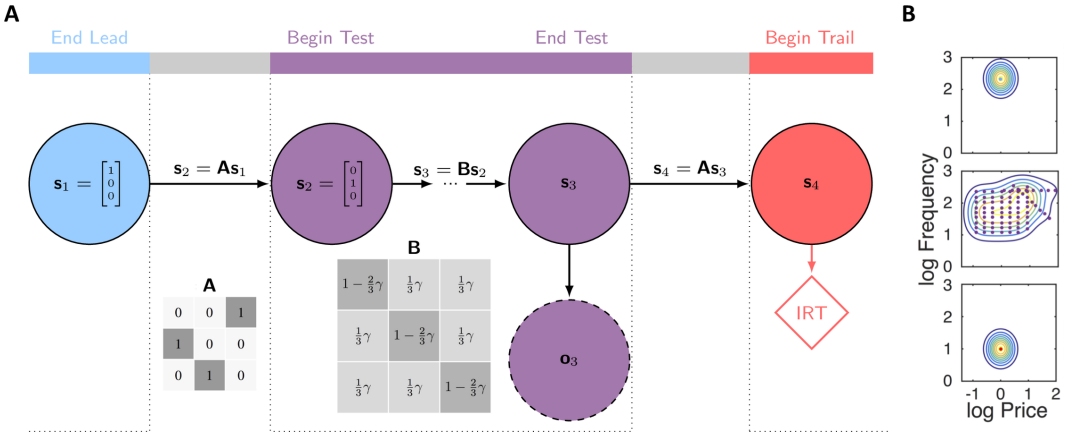

Figure 7.7: Modelling the inference process. (**A**) We characterise subjects as building an HMM generative model of the task and performing recognition to produce posterior subjective beliefs over the trial types. In our model, at the end of a lead trial the subject is certain it is on a lead trial (s_1). As it has learned the transition structure, described by matrix **A**, it is therefore certain it is on a test trial at the beginning of a test trial (s_2). If recognition was perfect, this knowledge would persist through the test trial; we model subjects' imperfection as arising from uncertainty in past evidence, which we describe using a parameter γ, which parameterises the matrix **B**. By the end of the test trial, past evidence is integrated with the within-trial evidence provided by observations (o_3) of frequency and price. This leads to a posterior belief (s_3), which then leads to the subjective belief about the trial type at the beginning of what is actually the trail trial (s_4). This can then be used to generate a response: either no engagement or engagement with an associated IRT. (**B**) We describe the association between points in frequency-price space and trial type using a mixture of Gaussians centered at the experimentally utilised points for lead (top), test (middle) and trail (bottom) trials. We introduce a standard deviation parameter (σ) which is shared across all points.

current state. This is well justified as trained subjects always responded rapidly and continually on lead trials, and always observed unambiguous evidence over its duration.

In our model, matrix **A** then operates on this belief such that the subject's certainty propagates into certainty that a test trial will come next. This is again supported by the fact that trained subjects responded reliably on test trials and with a distribution of IRTs different from those of lead and trail trials, indicating negligible confusion at this point.

If the subjects' inference was perfect relative to the actual Markov chain, they would continue to believe that they were in a test trial throughout its entirety. How-

ever, unlike other trials, during a test trial subjects may be presented with observations that are misleading as to the trial type. Continued belief therefore depends on subjects being able to correctly rely on past information in the face of competing and more recent evidence.

We model imperfections in the subjects' ability to do this as arising from an incorrect generative model involving an intermediate matrix (**B**). This allows for the possibility that subjects could switch their beliefs as to the trial type. Matrix **B** is parameterized by a scalar γ, which characterises the uncertainty in past evidence. If $\gamma = 1$, all trial types are *a priori* equally likely, and past evidence is completely ignored. If $\gamma = 0$, then there is no uncertainty, past evidence fully determines the inferred state and the test trial remains unambiguously known.

If the value of γ is intermediate, observations of frequency and price that resemble leading or trailing trials will license the potential for incorrect inference. There remains a question of how close the resemblance needs to be, i.e., the structure of the likelihood of observations given the underlying trial type. We introduce a further, standard deviation parameter (σ) that governs a kernel density likelihood estimate (mixture of Gaussians) in log frequency-price space. To specify the centre of each kernel or Gaussian, we use the real points in log frequency-price space experienced by each subject during the experiment, scaling the mixture weights in proportion to the number of times they were observed.

For a given triad of trials, probabilistic integration according to the HMM can be described using Bayes rule as:

$$\mathrm{P}(\mathrm{s}_3|\mathrm{o}_3, \mathrm{s}_2, \mathrm{b}(\mathrm{s}_2)) \propto \mathrm{P}(\mathrm{o}_3|\mathrm{s}_3) \sum_{\mathrm{s}_2 \in \{\mathrm{lead,test,trail}\}} \mathrm{P}(\mathrm{s}_3|\mathrm{s}_2)\mathrm{b}(\mathrm{s}_2) \tag{7.1}$$

where s_3 is the inferred trial type at the end of a test trial, o_3 is the observed frequency and price (f, P) on the test trial and $\mathrm{b}(\mathrm{s}_2)$ is the belief state at the beginning of the test trial. This inference is equivalent to the 'forward' part of the forward-backward algorithm, where $P(s_3|o_3, s_2, b(s_2))$ is the updated belief state. This is equivalent to the POMDP belief update (Section 1.4.3) when actions are unavail-

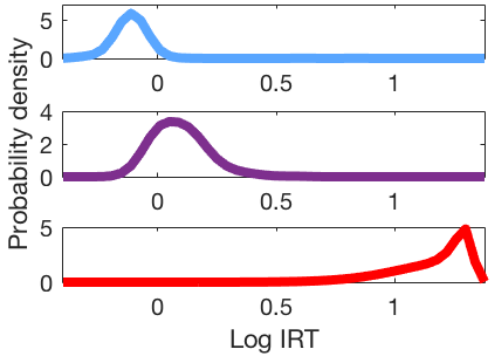

Figure 7.8: Determining the likelihood of responses given a posterior state.. We evaluated the probability of the observed responses given certainty about the trial type by constructing kernel density estimates of the observed responses. For lead and test trials, which do not lead to confusion, the density estimate was based directly on the observed distributions. For trail trials, to account for confusion, we first filtered the trials such that only those with preceding test trials in region λ were included.

able or irrelevant.

As, according to our model, the subject is certain it is on a test trial at the beginning of the test trial, our belief update simplifies to:

$$\mathrm{P}(\mathrm{s}_3|\mathrm{o}_3,\mathrm{s}_2) \propto \mathrm{P}(\mathrm{o}_3|\mathrm{s}_3)\mathrm{P}(\mathrm{s}_3|\mathrm{s}_2 = \mathrm{test}) \tag{7.2}$$

This makes clear the influence of both recent observations, $P(o_3|s_3)$, and evidence from the past, $P(s_3|s_2 = \mathrm{test})$, on the posterior belief at the end of a test trial. Having determined this belief we find the belief at the beginning of a trail trial, $\mathrm{P}(\mathrm{s}_4|\mathrm{o}_3,\mathrm{s}_2)$, simply by applying the task transition matrix $\mathbf{A}$ (marginalising out s_3).

We then calculate the probability of a particular response according to:

$$\mathrm{P}(\mathrm{r}_4|\mathrm{o}_3,\mathrm{s}_2) \propto \sum_{\mathrm{s}_4 \in \{\mathrm{lead,test,trail}\}} \mathrm{P}(\mathrm{r}_4|\mathrm{s}_4)\mathrm{P}(\mathrm{s}_4|\mathrm{o}_3,\mathrm{s}_2) \tag{7.3}$$

where r_4 is the response at the beginning of a trail trial (including no responses) and the summation is over the three possible trial types.

To calculate the probability of IRTs given a known trial type we used non-parametric fitting of lead, test and non-confusing trail trial IRTs (Fig 7.8).

The latter distribution was found by only selecting trail trials which followed

test trials in region λ, which thus largely eliminated short IRTs. In order to fit parameters γ and σ we used the real responses generated by the subjects and maximized the sum of the log likelihoods of those responses with respect to the parameters.

Having built the HMM we then split the data into three tertiles (details outlined in the following subsection), and determined the maximum likelihood estimate (MLE) of the parameters independently for each tertile. We were then able to simulate response distributions by sampling from $P(r_4^n | o_3^n, s_2^n = \text{test})$ where n indexes a particular triad of trials. We found that we were able to recover the pattern of short and long IRTs present in the real data, closely matching the observed distribution of EPs and IRTs for 5/6 subjects (Fig 7.9A;B;C). When sorting the simulated data by the previous test frequency and price in the same manner as before, the simulated data was found to match the real data well (Fig 7.9D), indicating that the model is able to account for the observed confusion.

To investigate simpler versions of the model that could provide a more parsimonious explanation for the observed responses we also tested models in which subjects only used evidence from one of frequency or price but not both, as well as models which either used past information perfectly ($\gamma = 0$) or not at all ($\gamma = 1$) (whilst using both frequency and price) (Table 7.1). These gave much poorer fits however as reflected by higher BIC scores, justifying the full version of the model over these alternatives. We also tested a more complex model, with an asymmetric matrix **B** due to parameters γ_f and γ_b which allow forward and backward transitions to be fit separately. This model was intended to test the hypothesis that subjects were more likely to prematurely transition their beliefs 'forwards' from test to trail rather than 'backwards' to lead. Interestingly, we found this asymmetry to be present, according to BIC, for two subjects. However, on average this model performed worse (by a score of 5.8) and so we do not use it for further analysis.

7.2.4 Inference improves with experience

Subjects typically encountered well over a thousand triads of trials. We therefore analysed improvements on the task with experience by dividing the data by triads

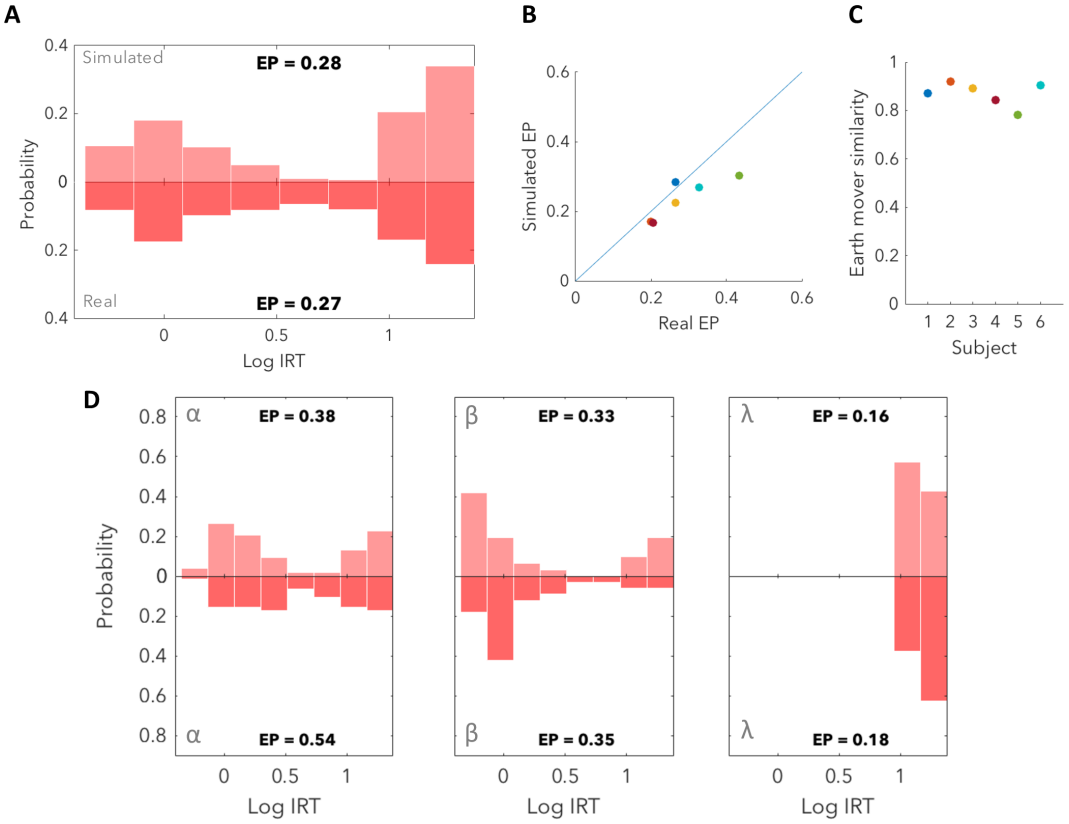

Figure 7.9: Simulated responses capture the process of mistaken inference. (**A**) By fitting model parameters and simulating responses (upper), we are able to recover the distribution of short and long IRTs observed in the data (lower)(subject 1). (**B**) Simulated EPs are similar to real EPs except for subject 5 (green). (**C**) The simulated distributions of IRTs have earth mover's similarities to the real distribution above 0.8 except for subject 5. (**D**) By sorting responses into regions as in Fig 7.4B. we find that simulated distributions are similar to the real distributions indicating that our model is able to capture the confusion effect.

Table 7.1: Relative increase in BIC score for alternative models

Subject	$\gamma = 0$	$\gamma = 1$	No f	No P	γ_f,γ_b
1	1331.5	261.9	15.2	30.4	14.4
2	173.8	33.7	26.8	33.1	11.4
3	115.5	37.7	9.6	6.1	-7.4
4	833.3	125.9	30.0	20.9	-4.1
5	400.7	63.1	28.6	76.3	2.8
6	822.0	70.6	47.8	70.0	17.7

into three sequential tertiles. When comparing the final tertile to the first tertile for subject 1 we observe a marked decrease both in the EP and in the probability of short IRTs on trail trials (Fig 7.10A). To analyse this across all subjects we calculated the

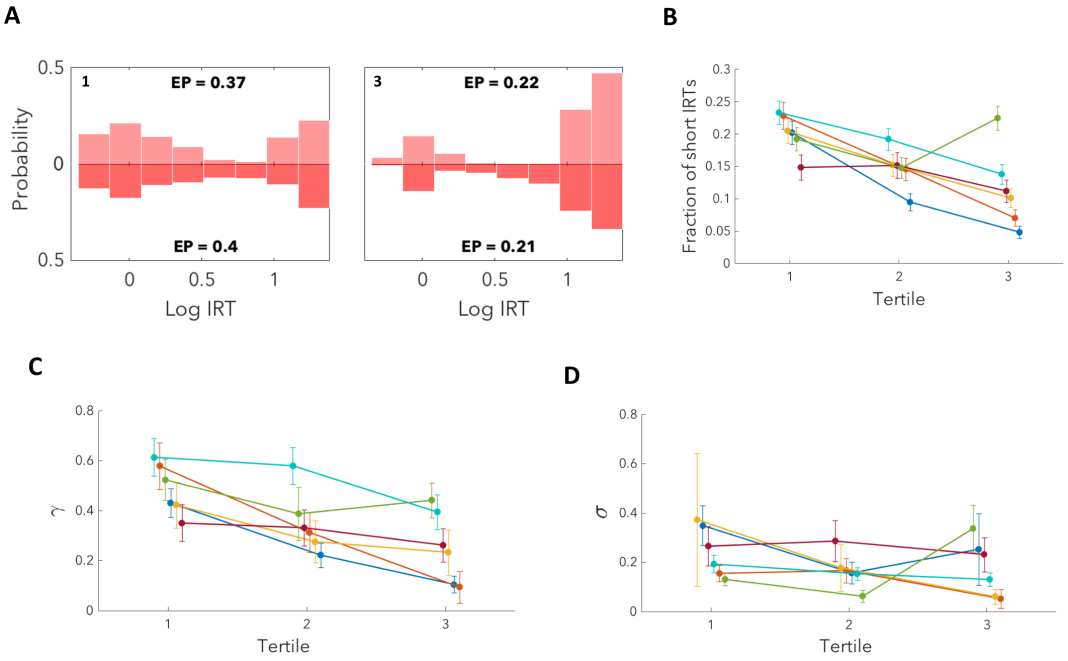

Figure 7.10: Mistaken inference becomes less likely with experience, as subjects learn to use past evidence. (**A**) Upper: simulated; lower: real. We divided the data into three tertiles, fit parameters independently for each tertile and simulated responses. We illustrate the first and last tertile in which subject 1 both lowers its EP and also decreases the probability of short IRTs in cases when it does respond. (**B**) By plotting the fraction of short IRTs in each tertile we find a significant decrease from first to last tertile for 4/6 subjects (permutation test; h_{11}) indicating that these subjects improve in their ability to identify the trail trial. The remaining two subjects show no significant change. (**C**) By calculating γ for each tertile we find that 4/6 subjects show a significant decrease in the MLE estimate of γ from the first to last tertiles, with the remaining two subjects showing no significant change (permutation test; h_{12}). Although significance was tested using a permutation test, we illustrate errorbars using the mean square error in the MLE of the parameters. The decrease in γ over time for the majority of subjects suggests a process by which subjects learn to use past evidence. (**D**) There is no significant change in the MLE of σ for 3/6 subjects with two subjects showing a significant decrease and one a significant increase (permutation test; h_{13}). We therefore do not find strong evidence to suggest that improvements in performance in the majority of subjects was due to a more accurate association of frequency and price with the trial type.

fraction of short IRTs for each tertile and found it to be significantly decreased for 4/6 subjects, with the remaining subjects showing no significant change (Fig 7.10B; permutation test; h_{11}). Taken together, this indicates that by the final tertile most subjects had improved their ability to track their progress through the task, as even on misleading trials they were rarely confused.

In order to understand these changes in the context of our model, we fit model parameters independently to each tertile. MLEs of the parameters identified significantly lower values of γ in the last tertile relative to the first tertile for 4/6 subjects, with the remaining subjects not showing a significant change (Fig 7.10C; permutation test; h_{12}). The subjects for which this parameter changed significantly corresponded to those which had shown a significant decrease in the fraction of short IRTs. This suggests that over time, the majority of the subjects learned to use evidence from the past more effectively and so improved their identification of the test and subsequent trail trials.

We also examined changes in the MLEs of the parameter σ across tertiles and found no significant change for 3/6 subjects, a significant decrease for two subjects and a significant increase for one subject (Fig 7.10D; permutation test; h_{13}). This indicates that for most subjects there is no evidence that improvements in performance can be attributed to a more accurate association of frequency and price with the appropriate trial type.

Finally, to assess the linear correlation between estimates of the parameters γ and σ we calculated the Pearson correlation coefficient from the negative inverse Hessian evaluated at the MLE (Fig 7.11; see Appendices B.5 for further details). We determined this coefficient separately for each subject and for each tertile, and typically found a negative value between -0.3 and -0.7, indicating moderate anticorrelation in the estimated parameters.

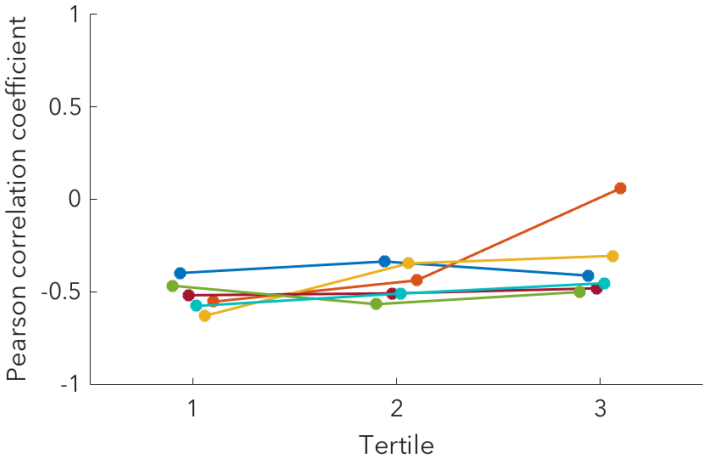

Figure 7.11: Estimates of the parameters γ and σ are moderately anticorrelated. We determined the linear correlation between estimates of the parameters γ and σ by calculating the Pearson correlation coefficient. We calculated this coefficient separately for each subject and for each tertile and typically found a negative value between -0.3 and -0.7, indicating moderate anticorrelation.

Chapter 8

Discussion

8.1 Findings and Limitations

We have shown that subjects learned a model of the world which reflected an experimentally defined transition structure. However, we also identified a small fraction of trials where behaviour seemingly went awry, as evidenced by subjects responding rapidly in advance of unrewarding trials. We demonstrated that these responses could be attributed to mistaken inference of the trial type, and described this process using an HMM. This involved introducing two parameters: σ, which influenced the mapping from observations during the trial to the inferred trial type; and γ, which represented the uncertainty associated with past evidence. We observed that γ decreased significantly over the course of hundreds of triads for the majority of subjects.

An important part of the work we have described is not only demonstrating subjects' abilities to learn structure in their environment but also in building a statistical model which describes inference in this context. The model developed was clearly defined and involved parameters which were interpretable, allowing for greater insight into the changing role of past and present evidence.

The representation used in our model proposed that subjects maintain belief states in an HMM. It is also conceivable that they might instead have adopted a less compressed, history-based representation of state, by storing the frequency and price of previous trials (either explicitly or implicitly). It is hard to distinguish these

based only on behaviour (particularly given the relative paucity of errors); but this would, of course, still constitute a functional form of world model.

In our preferred representation, observed 'mistakes', corresponding to short IRTs on worthless trail trials, are due to mistaken inference of the hidden state. To support this claim we demonstrated that when the inference problem was easy, such as following lead, trail or non-confusing test trials, subjects' responses reflected clear understanding of the structure. By contrast, when inference was hard, subjects more frequently responded inappropriately in a way which we were able to predict.

This imperfection arose in our model from uncertainty in past evidence, such that subjects failed to maintain their initial beliefs during a test trial. However, it is difficult to pin down the precise interpretation for this uncertainty. One interpretation is forgetting, or a lack of certainty in memory, which allows for a potential switching of beliefs when presented with observations which are more likely to have been generated by a lead or trail trial. Alternatively, this uncertainty could arise from imperfections in subjects' generative model, such that transitions could occur at any point during a trial as a result of misleading observations. One issue with this latter view is that subjects always experienced each trial as being deterministically stable across time, with no change in either price or frequency. Nevertheless, further work is necessary to distinguish between these two interpretations.

In our analysis we primarily focused on responses immediately after test trials, as only test trials varied across triads and thus posed substantial possibilities for confusion. By contrast, both lead and trail trials were unchanged in frequency, price and duration across triads, and empirically resulted in consistent response properties on subsequent trials after only a small amount of training. Whilst there may also have been confusion present early in training for these trials too, this learning may have progressed too rapidly to enable detailed analysis of its progress.

Our model is starkly simple, using only two parameters to predict behaviour without reference to the detailed microstructure of a given trial, such as the number of reward encounters or the average reward rate. Fig 7.6C provides a hint that the former factor can be influential, as increased sensitivity to price following high

frequency test trials may have resulted from an increased number of lever presses on these trials and thus a more accurate perception of price.

Nevertheless, we made this choice to capture and highlight the predominant effects observed across subjects whilst also maintaining interpretability. In turn, this allowed us to identify a significant change in the γ parameter in the majority of subjects, a finding supported both by calculating the standard error in the mean of these parameters and by permutation testing. In addition to identifying parameters we also determined linear correlations between them at the MLE, and found that estimates of γ and σ were moderately anticorrelated. This finding can be understood intuitively by considering a variation in the parameters such that γ is increased but σ is decreased. In this case, the uncertainty in past evidence increases but frequency and price are now more accurately perceived, resulting in a 'trade-off' between the two parameters when predicting behavioural performance. However, this trade-off is only partial, ultimately allowing for separate estimation of γ and σ with sufficient data.

Another related aspect of fitting our model was the choice not to use trial duration in addition to price to predict responses. As alluded to earlier, this was due to the strong correlation between duration and price, which implied that using either would produce similar results. On the other hand, we were able to show that subjects do use both frequency and price/duration, indicating that they successfully combined multiple sources of evidence in the inferential process.

Finally, it is interesting to consider the limits of rat intelligence. Whilst the task itself provided subjects with unsatiating reward which was clearly highly desirable, we found that they nevertheless remained somewhat absent-minded in their pursuit, even by the end of training. Furthermore, we found for all subjects that their IRTs on test trials were clearly distinguishable from that of lead trials, which would not be expected if they optimally balanced exploration with exploitation. Our results therefore highlight how models which assume optimal animal behaviour, may often fail to account for real world data.

8.2 Future work

8.2.1 Model learning

One avenue for future research would be to better understand how the subjects learned the overall model of the world over early training – particularly given their initially imperfect memories and their ignorance of the number of potential states. One promising approach is to consider a non-parametric statistical structure such as an infinite hidden Markov model (iHMM) (Beal et al., 2002). This formulation allows for countably infinite number of hidden states, in contrast to HMMs in which the number of hidden states is finite and specified in advance.

The iHMM uses the theory of Dirichlet processes to implicitly integrate out the infinitely many parameters, leaving just three hyperparameters which define the prior over transition dyanmics. These parameters are: an α parameter which controls the probability of self transitions and thus the time scale over which the dynamics of the hidden state evolves, a β parameter which controls the tendency of the model to populate previously untransitioned to states, and a γ parameter which controls the expected number of represented hidden states. The posterior over the hidden state sequence in an iHMM may be inferred using an approximate Gibbs sampling procedure.

Using an iHMM to model subjects' experiences early in training might therefore allow the growth in number of hidden states to be accurately inferred, as well as progress in subjects' understanding of the transition structure to be uncovered. It would be interesting to see if model building occurs during a series of 'Eureka!' moments or instead progresses more gradually. Likely there would be a degree of variability amongs subjects and this too would be interesting to explore.

8.2.2 Adaptive integration of past evidence

The task demands in excess of 50 seconds of memory in order for subjects to utilise information from two trials back. However, limits on the subjects' capacities, and the relationship between their willingness to deploy this expensive resource and the resulting distribution of rewards (Kurzban et al., 2013; Botvinick and Braver,

2015) are unclear. Unfortunately, the structure of the task made assessing the dynamics of memory within a trial difficult to uncover; whilst longer trials might be expected to result in more forgetting and decreased accuracy in our task, this effect is confounded by the ability of subjects to use an extended price/duration to infer trial type more accurately. It would therefore be worthwhile designing a task which decorrelates these variables, giving us a greater insight into the influence of time.

8.2.3 Neural underpinnings

Understanding the neural basis of the diverse processes involved in this task provides an exciting challenge for future research. In the case of working memory, its functioning is thought to be supported by persistent activity in a number of brain regions, including medial prefrontal cortex (Wang and Cai, 2006; Yoon et al., 2008; Horst and Laubach, 2009; Yang et al., 2014), entorinhal cortex (Hölscher and Schmidt, 1994; Egorov et al., 2002) and the hippocampus (Wang and Cai, 2006; Yoon et al., 2008). For evidence of neural representations of task structure, the hippocampus provides a natural candidate (Constantinescu et al., 2016; Garvert et al., 2017), and orbitofrontal cortex might similarly be suitable, given implications that it can encode a probability distribution over hidden causes (Wilson et al., 2014; Gershman et al., 2015; Schuck et al., 2016; Chan et al., 2016).

Epilogue

We explored two diverse topics involving both artificial and animal intelligence. Our findings in Part I demonstrated that a hierarchical organisation for interactions can enable reinforcement learning agents to cooperate more effectively to solve tasks, and can support scaling to larger problems. The choice of agent as manager was reflected in its ability to set goals for worker agents, an asymmetry which we found effective even in cases where communication of relevant information to the manager needed to be learned.

Our findings in Part II considered structure in a reward based task, and how this might be represented in the behaviour of rats. We constructed a model of each subject's predictions which involved it inferring the hidden task state and using this to respond accordingly. Our model was able to account for behavioural data, exploiting cases in which inference was mistaken to provide insight into the underlying algorithm and representation. By examining the slow change in behaviour over the course of extended training, we also gained a window onto improvements in the inference of subjects as they learned to integrate past evidence more accurately.

Although it is not a direction we were able to pursue directly, it is tempting to opine on the relationship between the two parts of the thesis. A perhaps superficial connection is the exploitation of different sorts of structure in the rather diverse tasks and architectures considered. In both parts we introduced externally imposed tasks with various demands and constraints, whether this be the differentiation between agents with information and agents who perform the task, or the rigid sequence of trial types for the rats. We then considered the internal mechanisms which could address them, whether this involved submission to feudal superiors or the allocation

of working memory.

A potentially deeper connection comes from considering the brain itself as a cooperating, multi-agent system. This organ is, after all, highly compartmentalised, with different regions performing different functions such as vision, memory and cognitive control. These systems must learn to communicate effectively to integrate their knowledge and must also jointly influence behaviour to increase an organism's chance for survival. It may therefore be possible for systems which learn according to Hebbian and affect-based principles to similarly provide each other with both instruction and reinforcement.

Appendix A

Appendix for Part I

A.1 Experimental results

A.1.1 Parameter settings for FMH

In all of our experiments in Chapters 3 and 4 we used the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 0.001 and $\tau = 0.01$ for updating the target networks. γ was 0.75. The size of the replay buffer was 10^7 and we updated the network parameters after every 100 samples added to the replay buffer. We used a batch size of 1024 episodes before making an update. For our feedforward networks we used two hidden layers with 256 neurons per layer in Chapter 3 but 64 neurons per hidden layer in Chapter 4. We trained with 10 random seeds (except otherwise stated). Error bars were plotted using the standard error in the mean in Chapter 3 and using the interquartile range in Chapter 4.

Hyperparameters were optimised using a line search centred on the experimental parameters used in Lowe et al. (2017). Our optimised parameters were found to be identical except for a lower value of γ (0.75) and of the learning rate (0.001), and a larger replay buffer (10^7). We found these values gave the best performance for both MADDPG and FMH on a version of Cooperative Communication with 6 landmarks evaluated after 50 epochs (an epoch is defined to be 1000 episodes).

A.1.2 Parameter sharing

We implemented parameter sharing for the decentralised algorithms DDPG and FMH. To make training results approximately similar to implementations which

do not use parameter sharing we restrict updates to a single agent and add experience only from a single agent to the shared replay buffer (amongst those sharing parameters). We find in practice that both approaches give very similar results for FMH, whereas parameter sharing slightly improves the performance of DDPG – we show this for a version of Cooperative Communication with 3 listeners and 6 targets (Figure A.1). In general sharing parameters reduces training time considerably, particularly as the number of agents scales.

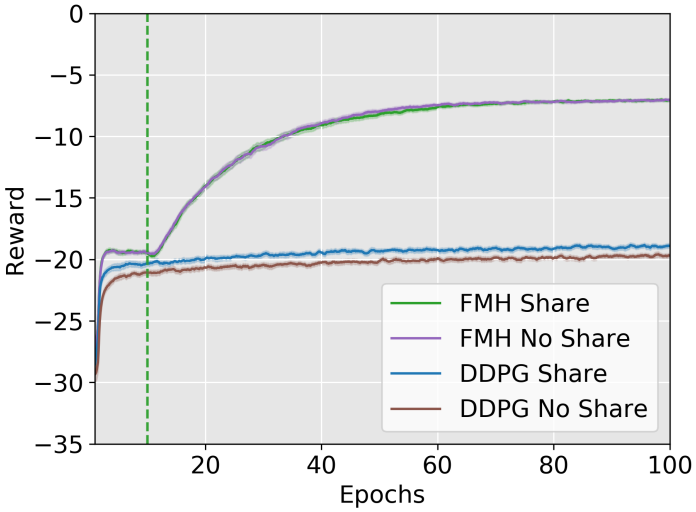

Figure A.1: Parameter sharing. Parameter sharing does not affect performance for FMH but slightly improves DDPG (Tensorflow).

A.1.3 Further details on Table 1

Values in the table were determined using 10 random seeds in all cases, except for the one exception of MADDPG with 10 listeners and 6 landmarks, which used 3 random seeds (training time is substantially longer as we do not share parameters). The CoM agent was trained on the synthetic task with different numbers of landmarks. Performance of the trained CoM policies was then evaluated over a period of 10 epochs on the corresponding true tasks.

Convergence was determined by comparing the mean performance in the final 5 epochs with the mean performance of a sliding window 5 epochs in width (we also take the mean across random seeds). If the mean performance within the window was within 2 percent of the final performance, and remained so for all subsequent

epochs, we defined this as convergence, unless the first time this happened was within the final 10 epochs. In such a case, we define the algorithm as not having converged. For assessing the exact time of convergence in the case of FMH we report values which include the 10 epochs of pretraining.

A.1.4 Cooperative communication with 3 landmarks

For reference we show performance of the various algorithms on Cooperative Communication with 3 landmarks in both Tensorflow and Pytorch. Both MADDPG and FMH perform well on this task, although MADDPG reaches convergence more rapidly (Figure A.2).

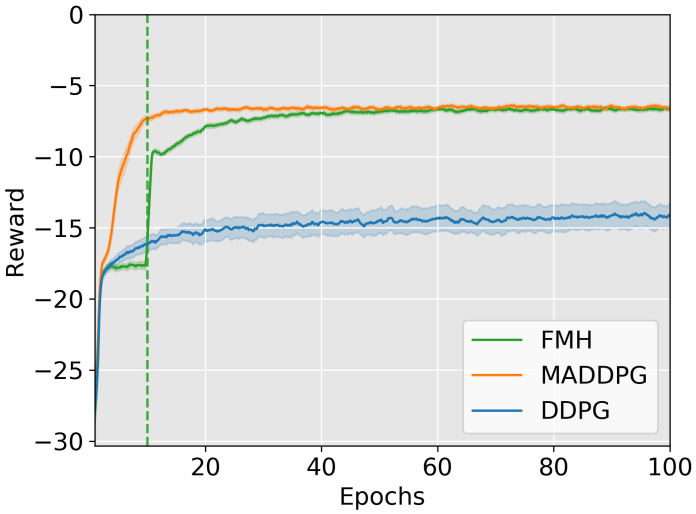

Figure A.2: Cooperative Communication with Tensorflow. 1 listener and 3 landmarks in Tensorflow implementation (n=10).

Our Tensorflow and Pytorch implementations achieve broadly similar results. Our Pytorch version also shows results for feudal MADDPG without communication repeats, which learns the fastest.

A.1.5 Differences in DDPG and MADDPG implementations

We use the implementation of DDPG in Chapter 3 provided by Lowe et al. (2017) [1]. In Chapter 4 we use DDPG and MADDPG as implemented by Iqbal and Sha (2019) [2]. One difference between the two is that Iqbal and Sha (2019) use the Straight-

[1] https://github.com/openai/maddpg
[2] https://github.com/shariqiqbal2810/maddpg-pytorch

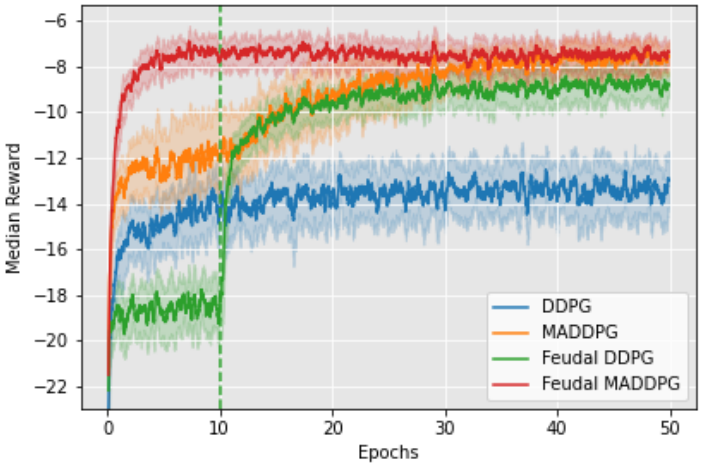

Figure A.3: Cooperative Communication with Pytorch. 1 listener and 3 landmarks in Pytorch implementation (n=4)

Through Gumbel-Softmax estimator which forces communication to be discrete on the forward pass. Another is that rather than sampling past actions from the replay buffer, they use the sampled observations to determine new actions using the current version of the policy. The updates for MADDPG are therefore:

$$\nabla_{\theta_i} J(\theta_i) = \mathbb{E}_{\boldsymbol{o}\sim\mathcal{D},\boldsymbol{a}\sim\boldsymbol{\mu}}[\nabla_{\theta_i}\mu_i(o_i)\nabla_{a_i}Q_i^{\boldsymbol{\mu}}(\boldsymbol{o},\boldsymbol{a})|_{a_i=\mu_i(o_i)}]. \tag{A.1}$$

$$\mathcal{L}(w_i) = \mathbb{E}_{\boldsymbol{o},\boldsymbol{o'}\sim\mathcal{D},\boldsymbol{a},\boldsymbol{a'}\sim\boldsymbol{\mu}}[(Q_i^w(\boldsymbol{o},\boldsymbol{a}) - y)^2] \tag{A.2}$$

where $y = r_i + \gamma Q_i(\boldsymbol{o'},\boldsymbol{a'})$. They also implement DDPG similarly.

Strictly speaking, this notation (also used by Iqbal and Sha (2019)) is not quite precise as the action is further transformed by a one-hot operation to mimic a sample from the replay buffer. Interestingly, we find that replacing this onehot operation instead with a soft Gumbel sample (temperature of 1) further improves the performance of DDPG, perhaps because it better reflects the current policy. We therefore use this in our experiments for both DDPG and MADDPG.

A.2 Environments

A.2.1 Cooperative communication

We provide further details on our version of Cooperative Communication (see main text for original description). In general, we keep environment details the same as Lowe et al., including the fact that the manager only has access to the target colour. However, we also scale up the number of coloured landmarks, which we do by taking the RGB values provided in the multi-agent particle environment, $(0.65, 0.15, 0.15), (0.15, 0.65, 0.15), (0.15, 0.15, 0.65)$, and adding 9 more by placing them on the remaining vertices of the corresponding cube and at the centre-point of four of the faces (in RGB space).

The particular colour values used for the landmarks influences the performance of RL algorithms as landmarks which have similar colours are harder for the speaker to learn to distinguish.

A.2.2 Cooperative coordination

We provide further details on our version of Cooperative Coordination (see main text for original description). The task provides a negative reward of -1 to each agent involved in a collision. For DDPG and MADDPG this penalty is shared across agents, whereas in FMH only the agents involved in the collision experience this penalty.

We also evaluated performance of trained policies in Figures 3.11c and 3.11d with slight modifications to the overall task. In the case of Figure 3.11c, to ensure that targets were never impossible to achieve by overlapping with the immobile manager, we moved the manager off-screen. For Figure 3.11d we ensured that agent positions were never initialised in a way such that they would automatically collide (such cases are rare).

A.2.3 Search and cooperative communication

We provide further details on our version of Search and Cooperative Communication (including v2, see main text for original description). In general, we keep the environment the same as in Cooperative Communication with 3 landmarks, but ad-

ditionally add an extra black landmark and allow each episode to run for 50 rather than 25 timesteps. In the original version of this task with 3 agents, the Information Gatherer may communicate to the Speaker two separate messages, each with 4 possible discrete values. The Speaker can communicate one discrete message to the Information Gatherer and one to the Listener (again 4 possible values for each). The Listener does not communicate at all. In v2 of the task there are only two agents and the Information Gatherer communicates a discrete message with 4 possible values directly to the Listener. The Listener is again silent.

A.2.4 Algorithm Specifics

We provide tables which summarise aspects of the different centralisation methods and feudal algorithms. In all cases communicated messages are received on the next time step (whether these be goals sent by the manager or regular messages sent by the workers).

Table A.1: Centralisation of all algorithms. Here P1 and P2 stand for Phases I and II.

Algorithm	Centralised Critics	Centralised Policies	No. actor-critics for N agents	Feudal
DDPG	No	No	N	No
MADDPG	Yes	No	N	No
FMH-DDPG	No	No	N	Yes
FMH-MADDPG	Yes	No	N	Yes
FMH-MADDPG-DDPG	Manager only	No	N	Yes
Single-agent CPAC	P1: Yes P2: Yes	P1: Yes P2: No	P1: 1 P2: N	P1: No P2: No
FMH-CPAC	P1: Yes P2: Yes	P1: Manager only P2: No	P1: N P2: N	P1: Yes P2: No

Table A.2: Feudal algorithms. Note, only FMH-DDPG uses pretraining and extended communication.

Algorithm	Message type	Goal set	Reward	Chapter
FMH-DDPG	Discrete	Target landmark	Negative dist.	3
FMH-MADDPG	Discrete	Target landmark	Negative dist.	4
FMH-MADDPG-DDPG	Discrete	Target landmark	Negative dist.	4
FMH Target State	Continuous	Target state	Negative dist.	5
FMH Target state change	Continuous	Target state change	Negative dist.	5
FMH Cosine	Continuous	Target direction	Cosine similarity	5

Appendix B

Appendix for Part II

This work was done in collaboration with the Shizgal lab, Concordia University. All procedures involving animals were carried out by the Shizgal lab, further details can be found in Solomon et al. (2017) and Ahilan et al. (2019).

Ethics Statement

Animal-care and experimental procedures were carried out in accordance with the principles in the Canadian Council on Animal Cares (CCAC) Guide to the Care and Use of Experimental Animals, with the approval of the Concordia University Animal Research Ethics Committee (certificate #: 30000302).

B.1 Analysis

We outline here elements of the modelling methodology. For a full description of the experimental methodology see Solomon et al. (2017).

Since all trial types terminate after set intervals (25s for lead and trail; a variable duration for the test), some care is necessary with the resulting censoring of the time during which the subjects could engage. Furthermore, we occasionally observed cases towards the end of the trail trial in which the subject briefly pressed the lever for such a short time that there was no possibility of obtaining reward. This might have been a Pavlovian reaction to the expectation of the upcoming lead trial.

To avoid problems from these cases, we counted a trial as having been engaged in for the purposes of the EP if at least one reward was obtained, and we only considered IRTs (defined as the time taken from the beginning of a trial to press

the lever for the first time) on those same trials. This constraint implies that initial responses after 24s on lead and trail trials would be impossible as there remains insufficient time to obtain a reward; so we only examine the properties of the IRT below this value. For test trials, which have variable trial duration, this ignored potential IRTs much larger than 24 seconds. However, in practice such cases were extremely rare (Fig 7.3A).

One facet of the experimental design is that the subjects received idiosyncratic calibrated frequencies of brain stimulation reward. We duly defined lower (l) and upper (u) boundaries of the regions α, β and λ separately for each animal; these also defined the boundaries dividing low, intermediate and high frequencies in Figure 4. Table B.1 summarises these values. All frequencies are in Hertz and all prices are in seconds.

Table B.1: Frequencies and prices used for lead and trail trials and for boundaries of regions α, β and λ

Subject	f_{lead}	P_{lead}	f_{trail}	P_{trail}	f_u^{β}	f_l^{α}	$P_l^{\beta/\alpha}$	$P_u^{\beta/\alpha}$	f_l^{λ}	f_u^{λ}	P_l^{λ}
1	217.4	1.0	10.0	1.0	20.0	125.9	0.4	3.9	31.6	79.4	8.1
2	196.1	1.0	10.0	1.0	35.5	100.0	0.4	3.9	63.1	125.9	4.1
3	250.0	1.0	10.0	1.0	44.7	158.5	0.4	3.9	79.4	158.5	4.1
4	200.0	1.0	10.0	1.0	31.6	100.0	0.4	3.9	50.1	79.4	4.1
5	250.0	1.0	10.0	1.0	28.2	125.9	0.4	3.9	39.8	100.0	4.1
6	163.9	1.0	10.0	1.0	25.1	79.4	0.4	3.9	39.8	100.0	4.1

B.2 Statistical tests

We tested for statistical significance using two-tailed permutation and binomial proportion tests. Permutation tests were used to determine the probability that the observed difference in the test statistics between classes would occur for class labels which were randomly permuted. In all cases we used 1000 simulations.

One non-trivial usage of the permutation test was to see if changes in the MLE of model parameters was significant. For this we determined the MLE of the model parameters in the first and last tertiles for data in which the time labels were permuted randomly and calculated the absolute difference between these parameter values. This was repeated 1000 times in order to generate a distribution of differences. We then tested if the absolute difference in the MLE of the parameters for

the non-permuted data was significant (greater than the 95th percentile).

The binomial proportion tests were used to determine the probability of the equality of two binomial proportions for two observed distributions. To compute this we evaluated the test statistic:

$$Z = \frac{\hat{p}_1 - \hat{p}_2}{\sqrt{\hat{p}(1-\hat{p})(\frac{1}{n_1} + \frac{1}{n_2})}} \tag{B.1}$$

where $\hat{p}_1$ and $\hat{p}_2$ are the empirical probabilities, n_1 and n_2 the corresponding number of observations and:

$$\hat{p} = \frac{n_1\hat{p}_1 + n_2\hat{p}_2}{n_1 + n_2} \tag{B.2}$$

We then calculated p-values from Z using the normal approximation.

B.3 Null hypotheses and p-values

Our null hypotheses referenced in the Results section were as follows:

h_1: Trail trial IRTs have the same median as a combined grouping of lead and test IRTs for untrained subjects (permutation test)

h_2: Test trial IRTs have the same median as lead trial IRTs for trained subjects (permutation test)

h_3: EPs on trail trials are the same for trained subjects as they are for untrained subjects (binomial proportion test)

h_4: Lead trial IRTs have the same median for trained subjects and untrained subjects (permutation test)

h_5: Test trial IRTs are equally similar to trail trial IRTs with preceding test trials in region α as lead trial IRTs (permutation test)

h_6: Lead trial IRTs are equally similar to trail trial IRTs with preceding test trials in region β as test trial IRTs (permutation test)

h_7: The fraction of short trail trial IRTs is the same for the 'Intermediate' category as for the 'Low f' category, with P = 1s (binomial proportion test)

h_8: The fraction of short trail trial IRTs is the same for the 'Intermediate' category as for the 'High f' category, with P = 1s (binomial proportion test)

h_9: The fraction of short trail trial IRTs is the same for the 'P = 1s' category as for the 'P $>$ 7s' category, with high f (binomial proportion test)

h_{10}: The fraction of short trail trial IRTs is the same for the 'Low f' category as for the 'High f' category, with P $<$ 0.3s (binomial proportion test)

h_{11}: The fraction of short trail trial IRTs is the same in the final tertile as it is in the first tertile (binomial proportion test)

h_{12}: The MLE of γ is the same in the final tertile as it is in the first tertile (permutation test)

h_{13}: The MLE of σ is the same in the final tertile as it is in the first tertile (permutation test)

The P-values for these hypotheses for all subjects are listed in Table B.2.

Table B.2: P-values for null hypotheses

Null hypothesis	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5	Subject 6
h_1	0.506	0.001	0.472	0.200	0.488	0.543
h_2	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
h_3	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
h_4	0.024	<0.001	0.532	0.025	<0.001	0.723
h_5	<0.001	<0.001	<0.001	0.003	0.003	0.001
h_6	0.023	0.181	<0.001	0.003	0.414	0.177
h_7	0.004	<0.001	<0.001	0.014	<0.001	<0.001
h_8	0.002	<0.001	<0.001	0.002	<0.001	<0.001
h_9	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
h_{10}	0.007	0.058	<0.001	<0.001	0.014	0.108
h_{11}	<0.001	<0.001	<0.001	0.165	0.210	<0.001
h_{12}	<0.001	<0.001	0.049	0.204	0.232	<0.001
h_{13}	0.355	0.044	0.065	0.743	<0.001	0.022

B.4 Model comparison

We calculate the Bayesian Information Criterion (BIC) for a given model M according to:

$$\mathrm{BIC} = -2\log\mathrm{P}(\mathrm{D}|\theta^{\mathrm{ML}},\mathrm{M}) + N_M log N_D \tag{B.3}$$

Where D is observed data, θ^{ML} are the maximum likelihood parameters of the model, N_M is the number of model parameters and N_D is the number of data points.

As we split the data into tertiles, we calculate the BIC for each tertile first and sum these to form an overall BIC for each subject.

B.5 Comparison of model parameters across tertiles

When comparing model parameters across tertiles, for illustration in Fig 7.10C;D, we determined the standard error in the MLE of the parameters $\theta = (\gamma, \sigma)$ according to:

$$SE(\theta_i^{ML}) = \sqrt{T_{ii}(\theta^{ML})} \tag{B.4}$$

where θ_i^{ML} is the MLE of the parameter in question, T_{ii} is the i$^{\text{th}}$ diagonal element of the matrix $T = -H^{-1}$, the negative inverse of the Hessian H, defined as:

$$H_{ij}(\theta^{ML}) = \frac{\partial^2}{\partial\theta_i\theta_j} l(\theta)\bigg|_{\theta^{ML}} \tag{B.5}$$

where $l(\theta)$ is the log likelihood.

As the matrix T is an estimator of the asymptotic covariance matrix we use it to determine the Pearson correlation coefficient:

$$\rho_{ij} = \frac{T_{ij}(\theta^{ML})}{\sqrt{T_{ii}(\theta^{ML})}\sqrt{T_{jj}(\theta^{ML})}} \tag{B.6}$$

$$\tag{B.7}$$

Bibliography

S Ahilan and P Dayan. Feudal multi-agent hierarchies for cooperative reinforcement learning. In *Workshop on Structure & Priors in Reinforcement Learning (SPiRL 2019) at ICLR 2019*, pages 1–11, 2019.

Sanjeevan Ahilan and Peter Dayan. Correcting experience replay for multi-agent communication. *arXiv preprint arXiv:2010.01192*, 2020.

Sanjeevan Ahilan, Rebecca B Solomon, Yannick-André Breton, Kent Conover, Ritwik K Niyogi, Peter Shizgal, and Peter Dayan. Learning to use past evidence in a sophisticated world model. *PLoS computational biology*, 15(6):e1007093, 2019.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Advances in Neural Information Processing Systems*, pages 5048–5058, 2017.

Dmitriy Aronov, Rhino Nevers, and David W Tank. Mapping of a non-spatial dimension by the hippocampal–entorhinal circuit. *Nature*, 543(7647):719–722, 2017.

Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Bowen Baker, Ingmar Kanitscheider, Todor Markov, Yi Wu, Glenn Powell, Bob McGrew, and Igor Mordatch. Emergent tool use from multi-agent autocurricula. *arXiv preprint arXiv:1909.07528*, 2019.

Gianluca Baldassarre and Marco Mirolli. *Intrinsically motivated learning in natural and artificial systems*. Springer, 2013.

Stefan Banach. Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales. *Fund. math*, 3(1):133–181, 1922.

Andrew G Barto and Sridhar Mahadevan. Recent advances in hierarchical reinforcement learning. *Discrete event dynamic systems*, 13(1-2):41–77, 2003.

Tobias Baumann, Thore Graepel, and John Shawe-Taylor. Adaptive mechanism design: Learning to promote cooperation. *arXiv preprint arXiv:1806.04067*, 2018.

Matthew J Beal, Zoubin Ghahramani, and Carl Edward Rasmussen. The infinite hidden markov model. *Advances in neural information processing systems*, 1: 577–584, 2002.

Timothy EJ Behrens, Timothy H Muller, James CR Whittington, Shirley Mark, Alon B Baram, Kimberly L Stachenfeld, and Zeb Kurth-Nelson. What is a cognitive map? organizing knowledge for flexible behavior. *Neuron*, 100(2):490–509, 2018.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009.

Dimitri P Bertsekas, Dimitri P Bertsekas, Dimitri P Bertsekas, and Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena scientific Belmont, MA, 1995.

Matthew Botvinick and Todd Braver. Motivation and cognitive control: from behavior to neural mechanism. *Annual Review of Psychology*, 66, 2015.

Craig Boutilier. Planning, learning and coordination in multiagent decision processes. In *Proceedings of the 6th conference on Theoretical aspects of rationality and knowledge*, pages 195–210. Morgan Kaufmann Publishers Inc., 1996.

Steven J Bradtke and Michael O Duff. Reinforcement learning methods for continuous-time markov decision problems. In *Advances in neural information processing systems*, pages 393–400, 1995.

Keller Breland and Marian Breland. The misbehavior of organisms. *American psychologist*, 16(11):681, 1961.

Yannick-André Breton. *Molar and molecular models of performance for rewarding brain stimulation*. PhD thesis, Concordia University, 2013.

Yannick-André Breton, James C Marcus, and Peter Shizgal. Rattus psychologicus: construction of preferences by self-stimulating rats. *Behavioural brain research*, 202(1):77–91, 2009.

Yannick-André Breton, Ada Mullett, Kent Conover, and Peter Shizgal. Validation and extension of the reward-mountain model. *Frontiers in behavioral neuroscience*, 7, 2013.

Lucian Busoniu, Robert Babuska, and Bart De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, And Cybernetics-Part C: Applications and Reviews, 38 (2), 2008*, 2008.

Stephanie CY Chan, Yael Niv, and Kenneth A Norman. A probability distribution over latent causes, in the orbitofrontal cortex. *Journal of Neuroscience*, 36(30): 7817–7828, 2016.

Yu-Han Chang, Tracey Ho, and Leslie P Kaelbling. All learning is local: Multi-agent learning in global reward games. In *Advances in neural information processing systems*, pages 807–814, 2004.

Gang Chen. A new framework for multi-agent reinforcement learning–centralized training and exploration with decentralized execution via policy distillation. *arXiv preprint arXiv:1910.09152*, 2019.

Nuttapong Chentanez, Andrew G Barto, and Satinder P Singh. Intrinsically motivated reinforcement learning. In *Advances in neural information processing systems*, pages 1281–1288, 2005.

Kamil Ciosek and Shimon Whiteson. Offer: Off-environment reinforcement learning. 2017.

John D Co-Reyes, YuXuan Liu, Abhishek Gupta, Benjamin Eysenbach, Pieter Abbeel, and Sergey Levine. Self-consistent trajectory autoencoder: Hierarchical reinforcement learning with trajectory embeddings. *arXiv preprint arXiv:1806.02813*, 2018.

Alexandra O Constantinescu, Jill X O'Reilly, and Timothy EJ Behrens. Organizing conceptual knowledge in humans with a gridlike code. *Science*, 352(6292):1464–1468, 2016.

Nathaniel D Daw, Yael Niv, and Peter Dayan. Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control. *Nature neuroscience*, 8(12):1704, 2005.

Nathaniel D Daw, Aaron C Courville, and David S Touretzky. Representation and timing in theories of the dopamine system. *Neural computation*, 18(7):1637–1677, 2006.

Peter Dayan. Feudal q-learning. *1995*, 1994.

Peter Dayan and Nathaniel D Daw. Decision theory, reinforcement learning, and the brain. *Cognitive, Affective, & Behavioral Neuroscience*, 8(4):429–453, 2008.

Peter Dayan and Geoffrey E Hinton. Feudal reinforcement learning. In *Advances in neural information processing systems*, pages 271–278, 1993.

Peter Dayan, Yael Niv, Ben Seymour, and Nathaniel D Daw. The misbehavior of value and the discipline of the will. *Neural networks*, 19(8):1153–1160, 2006.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

Thomas G Dietterich. Hierarchical reinforcement learning with the maxq value function decomposition. *Journal of Artificial Intelligence Research*, 13:227–303, 2000.

Alexei V Egorov, Bassam N Hamam, Erik Fransén, Michael E Hasselmo, and Angel A Alonso. Graded persistent activity in entorhinal cortex neurons. *Nature*, 420(6912):173–179, 2002.

Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. Reverse curriculum generation for reinforcement learning. *arXiv preprint arXiv:1707.05300*, 2017.

Jakob Foerster, Ioannis Alexandros Assael, Nando de Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 2137–2145, 2016.

Jakob N Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Michael J Frank, Bryan Loughry, and Randall C O'Reilly. Interactions between frontal cortex and basal ganglia in working memory: a computational model. *Cognitive, Affective, & Behavioral Neuroscience*, 1(2):137–160, 2001.

Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477*, 2018.

Joaquin M Fuster. Network memory. *Trends in neurosciences*, 20(10):451–459, 1997.

Mona M Garvert, Raymond J Dolan, and Timothy EJ Behrens. A map of abstract relational knowledge in the human hippocampal–entorhinal cortex. *eLife*, 6:e17086, 2017.

Samuel J Gershman, David M Blei, and Yael Niv. Context, learning, and extinction. *Psychological review*, 117(1):197, 2010.

Samuel J Gershman, Kenneth A Norman, and Yael Niv. Discovering latent causes in reinforcement learning. *Current Opinion in Behavioral Sciences*, 5:43–50, 2015.

Jan Gläscher, Nathaniel Daw, Peter Dayan, and John P O'Doherty. States versus rewards: dissociable neural prediction error signals underlying model-based and model-free reinforcement learning. *Neuron*, 66(4):585–595, 2010.

Paul Glasserman and Yu-Chi Ho. *Gradient estimation via perturbation analysis*, volume 116. Springer Science & Business Media, 1991.

Joshua I Gold and Michael N Shadlen. Banburismus and the brain: decoding the relationship between sensory stimuli, decisions, and reward. *Neuron*, 36(2):299–308, 2002.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control. *arXiv preprint arXiv:1611.07507*, 2016.

Thomas L Griffiths, Michael I Jordan, Joshua B Tenenbaum, and David M Blei. Hierarchical topic models and the nested chinese restaurant process. In *Advances in neural information processing systems*, pages 17–24, 2004.

Emil Julius Gumbel. Statistical theory of extreme values and some practical applications. *NBS Applied Mathematics Series*, 33, 1954.

Jayesh K Gupta, Maxim Egorov, and Mykel Kochenderfer. Cooperative multi-agent control using deep reinforcement learning. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 66–83. Springer, 2017.

Harry F Harlow. The formation of learning sets. *Psychological review*, 56(1):51, 1949.

Masahiko Haruno, Daniel M Wolpert, and Mitsuo Kawato. Mosaic model for sensorimotor learning and control. *Neural computation*, 13(10):2201–2220, 2001.

Serhii Havrylov and Ivan Titov. Emergence of language with multi-agent games: Learning to communicate with sequences of symbols. In *Advances in neural information processing systems*, pages 2149–2159, 2017.

Wayne A Hershberger. An approach through the looking-glass. *Animal Learning & Behavior*, 14(4):443–451, 1986.

Christian Hölscher and Werner J Schmidt. Quinolinic acid lesion of the rat entorhinal cortex pars medialis produces selective amnesia in allocentric working memory (wm), but not in egocentric wm. *Behavioural brain research*, 63(2): 187–194, 1994.

Nicole K Horst and Mark Laubach. The role of rat dorsomedial prefrontal cortex in spatial working memory. *Neuroscience*, 164(2):444–456, 2009.

Junling Hu, Michael P Wellman, et al. Multiagent reinforcement learning: theoretical framework and an algorithm. In *ICML*, volume 98, pages 242–250. Citeseer, 1998.

Leonid Hurwicz. The design of mechanisms for resource allocation. *The American Economic Review*, 63(2):1–30, 1973.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Shariq Iqbal and Fei Sha. Actor-attention-critic for multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 2961–2970, 2019.

Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. *arXiv preprint arXiv:1611.05397*, 2016.

Max Jaderberg, Valentin Dalibard, Simon Osindero, Wojciech M Czarnecki, Jeff Donahue, Ali Razavi, Oriol Vinyals, Tim Green, Iain Dunning, Karen Simonyan, et al. Population based training of neural networks. *arXiv preprint arXiv:1711.09846*, 2017.

Max Jaderberg, Wojciech M Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castaneda, Charles Beattie, Neil C Rabinowitz, Ari S Morcos, Avraham Ruderman, et al. Human-level performance in 3d multiplayer games with population-based reinforcement learning. *Science*, 364(6443):859–865, 2019.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Natasha Jaques, Angeliki Lazaridou, Edward Hughes, Caglar Gulcehre, Pedro Ortega, Dj Strouse, Joel Z Leibo, and Nando De Freitas. Social influence as intrinsic motivation for multi-agent deep reinforcement learning. In *International Conference on Machine Learning*, pages 3040–3049, 2019.

Michael C Jensen and William H Meckling. Theory of the firm: Managerial behavior, agency costs and ownership structure. *Journal of financial economics*, 3(4): 305–360, 1976.

Yiding Jiang, Shixiang Gu, Kevin Murphy, and Chelsea Finn. Language as an abstraction for hierarchical deep reinforcement learning. *arXiv preprint arXiv:1906.07343*, 2019.

Jiarui Jin, Ming Zhou, Weinan Zhang, Minne Li, Zilong Guo, Zhiwei Qin, Yan Jiao, Xiaocheng Tang, Chenxi Wang, Jun Wang, et al. Coride: Joint order dispatch-

ing and fleet management for multi-scale ride-hailing platforms. *arXiv preprint arXiv:1905.11353*, 2019.

Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101 (1):99–134, 1998.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

Vijay R Konda and John N Tsitsiklis. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014, 2000.

George Konidaris and Andrew G Barto. Building portable options: Skill transfer in reinforcement learning. In *IJCAI*, volume 7, pages 895–900, 2007.

Kai A Krueger and Peter Dayan. Flexible shaping: How learning in small steps helps. *Cognition*, 110(3):380–394, 2009.

Tejas D Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Advances in neural information processing systems*, pages 3675–3683, 2016.

Saurabh Kumar, Pararth Shah, Dilek Hakkani-Tur, and Larry Heck. Federated control with hierarchical multi-agent deep reinforcement learning. *arXiv preprint arXiv:1712.08266*, 2017.

Robert Kurzban, Angela Duckworth, Joseph W Kable, and Justus Myers. An opportunity cost model of subjective effort and task performance. *Behavioral and Brain Sciences*, 36(6):661–679, 2013.

Jean-Jacques Laffont and David Martimort. *The theory of incentives: the principal-agent model*. Princeton university press, 2009.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324, 1998.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521 (7553):436–444, 2015.

Hyun-Rok Lee and Taesik Lee. Improved cooperative multi-agent reinforcement learning algorithm augmented by mixing demonstrations from centralized policy. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1089–1098. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

Joel Z Leibo, Vinicius Zambaldi, Marc Lanctot, Janusz Marecki, and Thore Graepel. Multi-agent reinforcement learning in sequential social dilemmas. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pages 464–473. International Foundation for Autonomous Agents and Multiagent Systems, 2017.

Máté Lengyel and Peter Dayan. Hippocampal contributions to control: the third way. In *Advances in neural information processing systems*, pages 889–896, 2008.

Andrew Levy, Robert Platt, and Kate Saenko. Hierarchical reinforcement learning with hindsight. *arXiv preprint arXiv:1805.08180*, 2018.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez,

Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Alex Tong Lin, Mark J Debord, Katia Estabridis, Gary Hewer, and Stanley Osher. Cesma: Centralized expert supervises multi-agents. *arXiv preprint arXiv:1902.02311*, 2019.

Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine Learning Proceedings 1994*, pages 157–163. Elsevier, 1994.

Zheng Liu, Elisabeth A Murray, and Barry J Richmond. Learning motivational significance of visual cues for reward schedules requires rhinal cortex. *Nature neuroscience*, 3(12):1307, 2000.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems*, pages 6379–6390, 2017.

Ryan Lowe, Jakob Foerster, Y-Lan Boureau, Joelle Pineau, and Yann Dauphin. On the pitfalls of measuring emergent communication. *arXiv preprint arXiv:1903.05168*, 2019.

Jinming Ma and Feng Wu. Feudal multi-agent deep reinforcement learning for traffic signal control. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2020.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

Rajbala Makar, Sridhar Mahadevan, and Mohammad Ghavamzadeh. Hierarchical multi-agent reinforcement learning. In *Proceedings of the fifth international conference on Autonomous agents*, pages 246–253. ACM, 2001.

Hongzi Mao, Mohammad Alizadeh, Ishai Menache, and Srikanth Kandula. Resource management with deep reinforcement learning. In *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*, pages 50–56, 2016.

Laetitia Matignon, Guillaume J Laurent, and Nadine Le Fort-Piat. Independent reinforcement learners in cooperative markov games: a survey regarding coordination problems. *The Knowledge Engineering Review*, 27(1):1–31, 2012.

David Mguni, Joel Jennings, Sergio Valcarcel Macua, Emilio Sison, Sofia Ceppi, and Enrique Munoz de Cote. Coordinating the crowd: Inducing desirable equilibria in non-cooperative systems. *arXiv preprint arXiv:1901.10923*, 2019.

Yasushi Miyashita. Neuronal correlate of visual associative long-term memory in the primate temporal cortex. *Nature*, 335(6193):817–820, 1988.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

Shakir Mohamed and Danilo Jimenez Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In *Advances in neural information processing systems*, pages 2125–2133, 2015.

John Moody and Matthew Saffell. Learning to trade via direct reinforcement. *IEEE transactions on neural Networks*, 12(4):875–889, 2001.

Igor Mordatch and Pieter Abbeel. Emergence of grounded compositional language in multi-agent populations. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Ofir Nachum, Shane Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. *arXiv preprint arXiv:1805.08296*, 2018a.

Ofir Nachum, Shixiang Gu, Honglak Lee, and Sergey Levine. Near-optimal representation learning for hierarchical reinforcement learning. *arXiv preprint arXiv:1810.01257*, 2018b.

Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*, volume 99, pages 278–287, 1999.

Pekka Niemi and Risto Näätänen. Foreperiod and simple reaction time. *Psychological Bulletin*, 89(1):133, 1981.

Ritwik K Niyogi, Yannick-Andre Breton, Rebecca B Solomon, Kent Conover, Peter Shizgal, and Peter Dayan. Optimal indolence: a normative microscopic approach to work and leisure. *Journal of The Royal Society Interface*, 11(91):20130969, 2014a.

Ritwik K Niyogi, Peter Shizgal, and Peter Dayan. Some work and some play: Microscopic and macroscopic approaches to labor and leisure. *PLOS Comput Biol*, 10(12):e1003894, 2014b.

John O'keefe and Lynn Nadel. *The hippocampus as a cognitive map*. Oxford: Clarendon Press, 1978.

James Olds and Peter Milner. Positive reinforcement produced by electrical stimulation of septal area and other regions of rat brain. *Journal of comparative and physiological psychology*, 47(6):419, 1954.

Randall C O'Reilly and Michael J Frank. Making working memory work: a computational model of learning in the prefrontal cortex and basal ganglia. *Neural computation*, 18(2):283–328, 2006.

Liviu Panait and Sean Luke. Cooperative multi-agent learning: The state of the art. *Autonomous agents and multi-agent systems*, 11(3):387–434, 2005.

Peng Peng, Ying Wen, Yaodong Yang, Quan Yuan, Zhenkun Tang, Haitao Long, and Jun Wang. Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play starcraft combat games. *arXiv preprint arXiv:1703.10069*, 2017.

Roger Ratcliff and Jeffrey N Rouder. Modeling response times for two-choice decisions. *Psychological Science*, 9(5):347–356, 1998.

Peter H Rudebeck and Elisabeth A Murray. Dissociable effects of subtotal lesions within the macaque orbital prefrontal cortex on reward-guided behavior. *Journal of Neuroscience*, 31(29):10569–10578, 2011.

Ahmad EL Sallab, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. Deep reinforcement learning framework for autonomous driving. *Electronic Imaging*, 2017(19):70–76, 2017.

Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. The starcraft multi-agent challenge. *arXiv preprint arXiv:1902.04043*, 2019.

David EM Sappington. Incentives in principal-agent relationships. *Journal of economic Perspectives*, 5(2):45–66, 1991.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International Conference on Machine Learning*, pages 1312–1320, 2015.

Nicolas W Schuck, Ming Bo Cai, Robert C Wilson, and Yael Niv. Human orbitofrontal cortex represents a cognitive map of state space. *Neuron*, 91(6):1402–1412, 2016.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

Wolfram Schultz, Peter Dayan, and P Read Montague. A neural substrate of prediction and reward. *Science*, 275(5306):1593–1599, 1997.

Paul Seabright. Managing local commons: theoretical issues in incentive design. *Journal of economic perspectives*, 7(4):113–134, 1993.

David Silver. Lecture 3: Planning by dynamic programming. *Google DeepMind*, 2015.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *ICML*, 2014.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Burrhus Frederic Skinner. *The behavior of organisms: An experimental analysis*. BF Skinner Foundation, 1990.

RB Solomon, I Trujillo-Pisanty, K Conover, and P Shizgal. Psychophysical inference of frequency-following fidelity in the neural substrate for brain stimulation reward. *Behavioural brain research*, 292:327–341, 2015.

Rebecca Brana Solomon, Kent Conover, and Peter Shizgal. Valuation of opportunity costs by rats working for rewarding electrical brain stimulation. *PloS one*, 12(8):e0182120, 2017.

Alec Solway, Carlos Diuk, Natalia Córdova, Debbie Yee, Andrew G Barto, Yael Niv, and Matthew M Botvinick. Optimal behavioral hierarchy. *PLOS Comput Biol*, 10(8):e1003779, 2014.

Martin Stolle and Doina Precup. Learning options in reinforcement learning. In *International Symposium on abstraction, reformulation, and approximation*, pages 212–223. Springer, 2002.

Peter Stone and Manuela Veloso. Layered learning. In *European Conference on Machine Learning*, pages 369–381. Springer, 2000.

Sainbayar Sukhbaatar, Rob Fergus, et al. Learning multiagent communication with backpropagation. In *Advances in Neural Information Processing Systems*, pages 2244–2252, 2016.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. 2018.

Richard S Sutton, Doina Precup, and Satinder P Singh. Intra-option learning about temporally abstract actions. In *ICML*, volume 98, pages 556–564, 1998.

Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

Richard S Sutton, Joseph Modayil, Michael Delp, Thomas Degris, Patrick M Pilarski, Adam White, and Doina Precup. Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 761–768. International Foundation for Autonomous Agents and Multiagent Systems, 2011.

Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*, pages 330–337, 1993.

Michael T Todd, Yael Niv, and Jonathan D Cohen. Learning to use working memory in partially observable environments through dopaminergic reinforcement. In *Advances in neural information processing systems*, pages 1689–1696, 2009.

Edward C Tolman. Cognitive maps in rats and men. *Psychological review*, 55(4): 189, 1948.

Edward C Tolman, Benbow F Ritchie, and Donald Kalish. Studies in spatial learning. i. orientation and the short-cut. *Journal of experimental psychology*, 36(1): 13, 1946.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, and Koray Kavukcuoglu. Feudal networks for hierarchical reinforcement learning. In *International Conference on Machine Learning*, pages 3540–3549, 2017.

Alexander Sasha Vezhnevets, Yuhuai Wu, Remi Leblond, and Joel Leibo. Options as responses: Grounding behavioural hierarchies in multi-agent rl. *arXiv preprint arXiv:1906.01470*, 2019.

William Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of finance*, 16(1):8–37, 1961.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

Mark E Walton, Timothy EJ Behrens, Mark J Buckley, Peter H Rudebeck, and Matthew FS Rushworth. Separable learning systems in the macaque brain and the role of orbitofrontal cortex in contingent learning. *Neuron*, 65(6):927–939, 2010.

Gong-Wu Wang and Jing-Xia Cai. Disconnection of the hippocampal–prefrontal cortical circuits impairs spatial working memory performance in rats. *Behavioural brain research*, 175(2):329–336, 2006.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4): 279–292, 1992.

Christopher John Cornish Hellaby Watkins. Learning from delayed rewards. 1989.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Robert C Wilson, Yuji K Takahashi, Geoffrey Schoenbaum, and Yael Niv. Orbitofrontal cortex as a cognitive map of task space. *Neuron*, 81(2):267–279, 2014.

David H Wolpert and Kagan Tumer. Optimal payoff functions for members of collectives. In *Modeling complexity in economic and social systems*, pages 355–369. World Scientific, 2002.

Sheng-Tao Yang, Yi Shi, Qi Wang, Ji-Yun Peng, and Bao-Ming Li. Neuronal representation of working memory in the medial prefrontal cortex of rats. *Molecular brain*, 7(1):61, 2014.

Taejib Yoon, Jeffrey Okada, Min W Jung, and Jeansok J Kim. Prefrontal cortex and hippocampus subserve different components of working memory in rats. *Learning & memory*, 15(3):97–105, 2008.

Chongjie Zhang, Sherief Abdallah, and Victor Lesser. Integrating organizational control into multi-agent learning. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 757–

764. International Foundation for Autonomous Agents and Multiagent Systems, 2009.

Eric A Zilli and Michael E Hasselmo. The influence of markov decision process structure on the possible strategic use of working memory and episodic memory. *PloS one*, 3(7):e2756, 2008.