
Communicating via Markov Decision Processes

Samuel Sokota^{*1} Christian Schroeder de Witt^{*2} Maximilian Igl³ Luisa Zintgraf² Philip Torr²
Martin Strohmeier⁴ J. Zico Kolter^{1,5} Shimon Whiteson^{2,3} Jakob Foerster²

Abstract

We consider the problem of communicating exogenous information by means of Markov decision process trajectories. This setting, which we call a Markov coding game (MCG), generalizes both source coding and a large class of referential games. MCGs also isolate a problem that is important in decentralized control settings in which cheap-talk is not available—namely, they require balancing communication with the associated cost of communicating. We contribute a theoretically grounded approach to MCGs based on maximum entropy reinforcement learning and minimum entropy coupling that we call MEME. Due to recent breakthroughs in approximation algorithms for minimum entropy coupling, MEME is not merely a theoretical algorithm, but can be applied to practical settings. Empirically, we show both that MEME is able to outperform a strong baseline on small MCGs and that MEME is able to achieve strong performance on extremely large MCGs. To the latter point, we demonstrate that MEME is able to losslessly communicate binary images via trajectories of Cartpole and Pong, while simultaneously achieving the maximal or near maximal expected returns, and that it is even capable of performing well in the presence of actuator noise.

1. Introduction

This work introduces a novel problem setting called Markov coding games (MCGs). MCGs are two-player decentralized Markov decision processes (Oliehoek et al., 2016) that proceed in four steps. In the first step, one agent (called the sender) receives a special private observation (called the

message), which it is tasked with communicating. In the second step, the sender plays out an episode of a Markov decision process (MDP). In the third, the other agent (called the receiver) receives the sender’s MDP trajectory as its observation. In the fourth, the receiver estimates the message. The shared payoff to the sender and receiver is a weighted sum of the cumulative reward yielded by the MDP and an indicator specifying whether or not the receiver correctly decoded the message.

Among the reasons that MCGs are of interest is the fact that they generalize other important settings. The first of these is referential games. In a referential game, a sender attempts to communicate a message to a receiver using cheap talk actions—i.e., communicatory actions that do not have externalities on the transition or reward functions. Referential games have been a subject of academic interest dating back at least as far as Lewis’s seminal work *Convention* (Lewis, 1969). Since then, various flavors of referential games have been studied in game theory (Skyrms, 2010), artificial life (Steels, 2013), evolutionary linguistics (Smith, 2002), cognitive science (Spike et al., 2017), and machine learning (Lazaridou et al., 2018). MCGs can be viewed as a generalization of referential games to a setting in which the sender’s actions may incur costs.

A second problem setting generalized by MCGs is source coding (MacKay, 2002) (also known as data compression). In source coding, the objective is to construct an injective mapping from a space of messages to the set of sequences of symbols (for some finite set of symbols) such that the expected output length is minimized. Source coding has a myriad of real world applications involving the compression of images, video, audio, and genetic data. MCGs can be viewed as a generalization of the source coding problem to a setting where the cost of an encoding may involve complex considerations, rather than simply being equal to the sequence length.

Yet another reason to be interested in MCGs is that they isolate an important subproblem of decentralized control. In particular, achieving good performance in an MCG requires the sender’s actions to simultaneously perform control in an MDP and communicate information (i.e., to communicate implicitly). This presents a challenge for approximate

^{*}Equal contribution ¹Carnegie Mellon University ²Oxford University ³Waymo Research ⁴armasuisse Science + Technology ⁵Bosch Center for AI. Correspondence to: Samuel Sokota <ssokota@andrew.cmu.edu>, Christian Schroeder de Witt <schroederdewitt@gmail.com>.

dynamic programming, the foundation for preeminent approaches to constructing control policies (Sutton & Barto, 2018), because the values of communication protocols depend on counterfactuals, violating dynamic programming’s locality assumption.

To address MCGs, we propose a theoretically grounded algorithm called MEME. MEME leverages a union of maximum entropy reinforcement learning (MaxEnt RL) (Ziebart et al., 2008) and minimum entropy coupling (MEC) (Kovačević et al., 2015). The key insight is that maximizing the returns of the MDP can be disentangled from learning a good communication protocol by means of observing that the entropy of a policy corresponds (in an informal sense) to its capacity to communicate. MEME leverages this insight in two steps. In the first step, MEME constructs a MaxEnt policy for the MDP, balancing between maximizing expected return and maximizing cumulative conditional entropy. This MaxEnt policy is provided to both the sender and the receiver. In the second step, at each decision point, MEME uses recent breakthroughs in MEC approximation techniques (Cicalese et al., 2019) to pair messages with actions in such a way that the sender selects actions with the same probabilities as the MaxEnt RL policy (thereby guaranteeing the same expected return from the MDP) and the receiver’s uncertainty about the message is greedily reduced as much as possible. After the sender completes its MDP episode, the receiver uses the MaxEnt RL policy and the sender’s trajectory to compute the posterior over the message and guesses the maximum a posteriori.

To demonstrate the efficacy of MEME, we present experiments for MCGs based on a gridworld, Cartpole, and Pong (Bellemare et al., 2013), which we call CodeGrid, CodeCart, and CodePong, respectively. For CodeGrid, we show that with a message space cardinality in the 10s or 100s, MEME significantly outperforms a baseline combining a reinforcement learning with an optimal receiver. For CodeCart and CodePong, we consider a message space of binary images and a uniform distribution over messages, meaning that a randomly guessing receiver has an astronomically small probability of guessing correctly. Remarkably, we show that MEME is able to achieve an optimal expected return in Cartpole and Pong while simultaneously losslessly communicating images to the receiver, demonstrating that MEME has the capacity to be scaled to extremely large message spaces and complex control tasks. Moreover, we find that the performance of MEME decays gracefully as the amount of actuator noise in the environment increases.

2. Related Work

The works that are most closely related to this one can be taxonomized as coming from literature on multi-agent reinforcement learning, coding, and diverse skill learning.

Multi-Agent Reinforcement Learning One body of related work investigates multi-agent reinforcement learning settings in which the agents are endowed with explicit communication channels. Foerster et al. (2016) propose DIAL, which optimizes the sender’s protocol by performing gradient ascent through the parameters of the receiver; Sukhbaatar et al. (2016) propose CommNet, which uses mean-pooling to process messages to handle dynamic number of agents over multiple steps; Das et al. (2019) propose TarMAC, which also handles targeted messaging; Mao et al. (2020a); Wang et al. (2020); Mao et al. (2020b) investigate handling settings in which the bandwidth of the communication channel is limited. Bhatt & Buro (2021) propose an approach for discrete communicate channels in which the sender’s behavioral policy deterministically selects the action that maximizes the receiver’s posterior probability of the correct message, when computed using the target policy. The problem setting we examine differs from this line of work in that the communication channel is not costless—instead, it possesses costs dictated by an MDP. More similar to our work in this respect is that of Strouse et al. (2018), who investigate auxiliary rewards for taking actions with high mutual information. The baseline for our CodeGrid experiments loosely resembles Strouse et al.’s algorithm.

Coding Another body of related work concerns extensions of the source coding problem. Length limited coding (Larimore & Hirschberg, 1990; Kostina & Verdú, 2011) considers a problem setting in which the objective is to minimize the expected sequence length (as before), subject to a maximum length constraint. Coding with unequal symbol costs (Golin et al., 2002; Iwata et al., 1997) considers the problem in which the goal is to minimize the expected cumulative symbol cost of the sequence to which the message is mapped. The cost of a symbol may differ from the cost of other symbols arbitrarily, making it a strictly more general problem setting than standard source coding (which can alternatively be thought of as minimizing cumulative symbol cost with equally costly symbols). Both length limited coding and coding with unequal costs are subsumed by Markov coding games. And while existing algorithms for both standard source coding and the extensions above are well-established and widely commercialized, they do not address the more general MCG setting.

MCGs are also related to finite state Markov channel settings (Wang & Moayeri, 1995) and Markov channel settings with feedback (Chen & Berger, 2005; Tatikonda & Mitter, 2009). In such settings, the fidelity of the channel by which the sender communicates to a receiver is controlled by a Markov process. Another related setting is intersymbol interference, where the sender’s previously selected symbols (i.e., actions) may cause interference with subsequently selected symbols, making them less likely to be faithfully transmitted to the

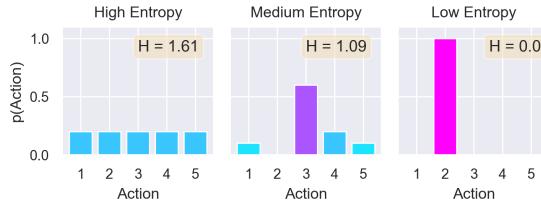


Figure 1. Three probability distributions over actions, in order of decreasing entropy (left to right).

receiver (Lathi, 1998). MCGs differ from both Markov channel and intersymbol interference settings in that the Markov system controls the cost of the channel, rather than its transmission quality. MCGs are more resemblant of a setting in which the channel is reliable, but subject to natural variation in costs, such as based on weather or third party usage, as well as variation based on the sender’s own usage.

Diverse Skill Learning A third area of related research is that of diverse skill learning (Eysenbach et al., 2018). Eysenbach et al. (2018) propose an unsupervised learning method for discovering diverse, identifiable skills. Their objective, called DIAYN, seeks to learn diverse, discriminable skills. This paradigm resembles our work in the sense that skills can be interpreted as messages and discriminability can be interpreted as maximizing the mutual information between the skill and the state. The baseline used in our CodeGrid experiments can also be viewed as an adaptation of an idealized version of DIAYN to the MCG setting.

3. Background and Notation

We will require the following background material.

Markov Decision Processes To represent our task formalism, we use finite Markov decision processes (MDPs). We denote MDPs using tuples $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{T} \rangle$ where \mathcal{S} is the set of states, \mathcal{A} is the set of actions, $\mathcal{R}: \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is the reward function, and $\mathcal{T}: \mathcal{S} \times \mathcal{A} \rightarrow \Delta(\mathcal{S})$ is the transition function. An agent’s interactions with an MDP are dictated by a policy $\pi: \mathcal{S} \rightarrow \Delta(\mathcal{A})$ mapping states to distributions over actions. We focus on episodic MDPs, meaning that after a finite number of transitions have occurred, the MDP will terminate. The history of states and actions is denoted using $h = (s^0, a^0, \dots, s^t)$, or $H = (S^0, A^0, \dots, S^t)$, if it is random. We use the notation $\mathcal{R}(h) = \sum_j \mathcal{R}(s^j, a^j)$ to denote the amount of reward accumulated over the course of a history. When a history is terminal, we use z (or Z , if it is random) to denote it, rather than h . The objective of an MDP is to construct a policy yielding a large cumulative reward in expectation $\mathbb{E}[\mathcal{R}(Z) | \pi]$.

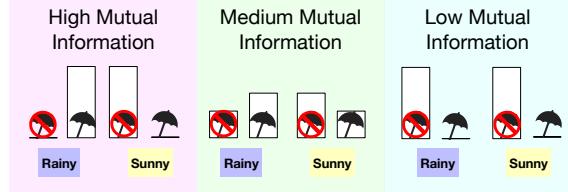


Figure 2. The mutual information between whether it rains and whether Alice (pink), Bob (green), and Charlie (teal) bring umbrellas, respectively. Alice always checks an unrealistically reliable weather station before leaving, so there is perfect mutual information. Bob makes a guess as to whether it will rain based on whether it is cloudy, so there is some mutual information. Charlie does not mind being rained on, so there is no mutual information.

Entropy To help us quantify the idea of uncertainty, we use entropy. The entropy of a random variable X is $\mathcal{H}(X) = -\mathbb{E} \log \mathcal{P}(X)$. It is maximized when the mass of \mathcal{P}_X is spread as evenly as possible and minimized when the mass of \mathcal{P}_X is concentrated at a point.

In the context of decision-making, entropy can be used to describe the uncertainty regarding which action will be taken by an agent (see Figure 1). When a policy spans multiple decision points, the uncertainty regarding the agent’s actions given that the state is known is naturally described by conditional entropy. Conditional entropy is the entropy of a random variable, conditioned upon the fact that the realization of another random variable is known. More formally, conditional entropy is defined by $\mathcal{H}(X | Y) = \mathcal{H}(X, Y) - \mathcal{H}(Y)$ where the joint entropy $\mathcal{H}(X, Y) = -\mathbb{E} \log \mathcal{P}(X, Y)$ is defined as the entropy of (X, Y) considered as a random vector.

In some contexts, it is desirable for a decision-maker’s policy to be highly stochastic. In such cases, an attractive alternative to the expected cumulative reward objective is the maximum entropy RL objective (Ziebart et al., 2008) $\max_{\pi} \mathbb{E}_{\pi} [\sum_t \mathcal{R}(S^t, A^t) + \alpha \mathcal{H}(A^t | S^t)]$, which trades off between maximizing expected return and pursuing trajectories along which its actions have large cumulative conditional entropy, using the temperature hyperparameter α .

Mutual Information A closely related concept to entropy is mutual information. Mutual information describes the strength of the dependence between two random variables. The greater the mutual information between two random variables, the more the outcome of one affects the conditional distribution of the other (see Figure 2). Symbolically, mutual information is defined by $\mathcal{I}(X; Y) = \mathcal{H}(Y) - \mathcal{H}(Y | X) = \mathcal{H}(X) - \mathcal{H}(X | Y)$. From this definition, we see explicitly that the mutual information of two random variables can be interpreted as the amount of uncertainty about one that is eliminated by observing the realization of the other.

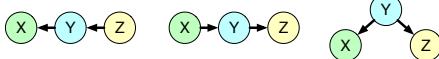


Figure 3. If any (or equivalently, all) of the graphs above are valid structures for a probabilistic graphical model for the joint distribution over X, Y, Z , and there are no confounding variables, the data processing inequality states that the dependence between X and Y must be at least as strong as that of X and Z .

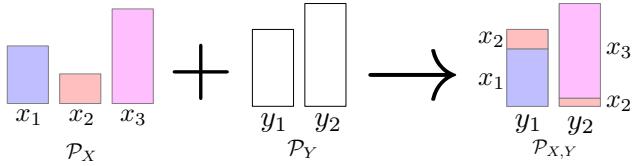


Figure 4. An approximate minimum entropy coupling. Given marginal distributions \mathcal{P}_X and \mathcal{P}_Y , minimum entropy coupling constructs a joint distribution $\mathcal{P}_{X,Y}$ having minimal joint entropy.

Mutual information is important for communication because we may only be able to share the realization of an auxiliary random variable, rather than that of the random variable of interest. In such cases, maximizing the amount of communicated information amounts to maximizing the mutual information between the auxiliary random variable and the random variable of interest.

The Data Processing Inequality The independence relationships among random variables play an important role in determining their mutual information. If random variables X and Z are conditionally independent given Y (that is, $X \perp Z | Y$), the data processing inequality states that $I(X; Y) \geq I(X; Z)$. Less formally, the inequality states that if Z does not provide additional information about X given Y , then the dependence between X and Z cannot be stronger than the dependence between X and Y . The conditional independence required for the data processing inequality is depicted visually in Figure 3.

Minimum Entropy Coupling In some cases, we may wish to maximize the mutual information between two random variables subject to fixed marginals. That is, we are tasked with determining a joint distribution $\mathcal{P}_{X,Y}$ that maximizes the mutual information $I(X; Y)$ between X and Y subject to the constraints that $\mathcal{P}_{X,Y}$ marginalizes to \mathcal{P}_X and \mathcal{P}_Y , where \mathcal{P}_X and \mathcal{P}_Y are given as input. Invoking the relationship between mutual information and joint entropy $I(X; Y) = H(X) + H(Y) - H(X, Y)$, we see that this problem is equivalent to that of minimizing the joint entropy of X and Y . As a result, this problem is referred to as the *minimum entropy coupling problem*. A visual example is shown in Figure 4. While minimum entropy coupling is NP-hard (Kovačević et al., 2015), Cicalese et al. (2019) recently



Figure 5. **A depiction of the structure of MCGs.** First, the sender is given a message. Second, the sender is tasked with a MDP, unrelated to the message. Third, the receiver observes the sender’s trajectory. Fourth, the receiver estimates the message.

showed that there exists a polynomial time algorithm that is suboptimal by no more than one bit.

4. Markov Coding Games

We are now ready to introduce Markov coding games (MCGs). An MCG is a tuple $\langle (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}), \mathcal{M}, \mu, \zeta \rangle$, where $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R})$ is a Markov decision process, \mathcal{M} is a set of messages, μ is a distribution over \mathcal{M} (i.e., the prior over messages), and ζ is a non-negative real number we call the message priority. **An MCG proceeds in the following steps:**

1. First, a message $M \sim \mu$ is sampled from the prior over messages and revealed to the sender.
2. Second, the sender uses a message conditional policy $\pi_{|M}$, which takes states $s \in \mathcal{S}$ and messages $m \in \mathcal{M}$ as input and outputs distributions over MDP actions $\Delta(\mathcal{A})$, to generate a trajectory $Z \sim (\mathcal{T}, \pi_{|M})$ from the MDP.
3. Third, the sender’s terminal MDP trajectory Z is given to the receiver as an observation.
4. Fourth, the receiver uses a terminal MDP trajectory conditional policy $\pi_{|Z}$, which takes terminal trajectories $z \in \mathcal{Z}$ as input and outputs distributions over messages $\Delta(\mathcal{M})$, to estimate the message $\hat{M} \sim \pi_{|Z}(Z)$.

The objective of the agents is to maximize the expected weighted sum of the return and the accuracy of the receiver’s estimate $\mathbb{E} [\mathcal{R}(Z) + \zeta \mathbb{I}[M = \hat{M}] | \pi_{|M}, \pi_{|Z}]$. Optionally, in cases in which a reasonable distance function is available, we allow for the objective to be modified to minimizing the distance between the message and the guess $d(M, \hat{M})$, rather than maximizing the probability that the guess is correct. A visual depiction of the structure MCGs is given in Figure 5.

4.1. An Example

A payoff matrix for a simple MCG is shown in Table 1. In this game, the sender is given one of two messages m and m' , with equal probability. It then chooses among three actions a, a' and a'' , for which the rewards are 4, 3, and 0, respectively. The receiver observes the sender’s action (which is equivalent to the sender’s action in MDPs with one state) and estimates the message using actions \hat{m} and

Table 1. A payoff matrix for a simple MCG.

Message	Sender	Receiver	
		\hat{m}	\hat{m}'
m	a	(4, ζ)	(4, 0)
	a'	(3, ζ)	(3, 0)
	a''	(0, ζ)	(0, 0)
m'	a	(4, 0)	(4, ζ)
	a'	(3, 0)	(3, ζ)
	a''	(0, 0)	(0, ζ)

\hat{m}' , corresponding to guessing to m and m' , respectively. The receiver accrues a reward of ζ for guessing correctly. The payoff entries in table denote $(\mathcal{R}(Z), \zeta \mathbb{I}[M = \hat{M}])$ for each outcome.

As is generally true of MCGs, this MCG is difficult for independent approximate dynamic programming-based approaches because their learning dynamics are subject to local optima. Consider a run in which the sender first learns to maximize its immediate reward by always selecting a . Now, the receiver has no reason to condition its guess on the sender’s action because the sender is not communicating any information about the message. As a result, thereafter, the sender has no incentive to communicate any information in its message, because the receiver would ignore it anyways. This outcome, sometimes called a babbling equilibrium (Farrell & Rabin, 1996), leads to a total expected return of $4 + \zeta/2$ (sender always selects a , receiver guesses randomly). In cases in which the message priority ζ is small (i.e., communication is unimportant), the babbling equilibrium performs well. However, it can perform arbitrarily poorly as ζ becomes large.

4.2. Special Cases

We can express both referential games and various source coding settings as special cases of the MCG formalism by describing the MDPs to which they correspond.

(A Large Class of) Referential Games We can express a T step referential game as follows.

- The state space $\mathcal{S} = \{s^0, s^1, \dots, s^T\}$.
- The transition function deterministically maps $s^t \mapsto s^{t+1}$ and terminates at input s^T .
- The reward maps to zero for every state action pair.

Standard Source Coding We can express the standard source coding problem as follows.

- The state space $\mathcal{S} = \{s\}$.
- The action space $\mathcal{A} = \tilde{\mathcal{A}} \cup \{\emptyset\}$.
- The transition function deterministically maps to s to s for all $a \in \tilde{\mathcal{A}}$ and terminates the game on \emptyset .
- The reward maps to -1 for $a \in \tilde{\mathcal{A}}$ and maps \emptyset to 0.

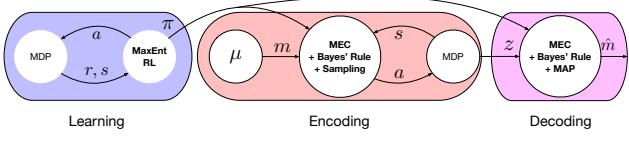


Figure 6. A high level depiction of MEME.

Length Limited Source Coding Length limited source coding can be captured in the same way as standard source coding with the modifications that $\mathcal{S} = \{s^0, s^1, \dots, s^T\}$, and the transition function maps $s^t \mapsto s^{t+1}$ and terminates on s^T , where T is the length limit.

Source Coding with Unequal Symbol Costs Source coding with unequal symbol costs can be captured in the same way as standard source coding with the modification that $\mathcal{R}(\cdot, a)$ returns the negative symbol cost of a , rather than returning -1 , for $a \in \mathcal{A}$.

5. Maximum Entropy Reinforcement Learning and Minimum Entropy Coupling

To address MCGs, we propose a novel algorithm we call MEME. MEME (and more broadly, any algorithm geared toward MCGs) is faced with two competing incentives. On one hand, it needs to maximize expected return $\mathcal{R}(Z)$ generated by the MDP trajectory Z . On the other hand, it needs to maximize the amount of information $\mathcal{I}(M; Z)$ communicated to the receiver about the message, so as to maximize the probability of a correct guess. MEME handles this trade-off using a two step process for constructing the sender’s policy. In the first step, it computes an MDP policy that balances between high cumulative reward and large cumulative entropy using MaxEnt RL. In the second step, it opportunistically leverages the existing uncertainty in the MaxEnt RL policy by coupling its probability mass with the posterior over messages in such a way that the expected return does not decrease and the amount of mutual information between the message and trajectory is greedily maximized. Finally, because the receiver is also privy to the MaxEnt RL policy, it can compute the exact same posterior over the message as the sender and guess the most likely message. This process is shown in Figure 6 and described in detail below. Thereafter, we show this procedure possesses desirable guarantees and discuss intuition and scalability.

5.1. Method Description (MEME)

Step One: Maximum Entropy Reinforcement Learning

In the first step, MEME uses MaxEnt RL to construct an MDP policy π . This policy is an MDP policy, not an MCG policy, and therefore does not depend on the message. Note

that this policy depends on the choice of temperature α used for the MaxEnt RL algorithm.

Step Two: Minimum Entropy Coupling In the second step, at execution time, MEME constructs a message-conditional policy online using MECs, as shown in Algorithm 1. Say that, up to time t , the sender is in state s^t , history h^t and has played according to the state and message conditional policy $\pi_{|M}^{:t}$ so far. Let

$$b^t = \mathcal{P}(M | h^t, \pi_{|M}^{:t})$$

be the posterior over the message, conditioned on the history and the historical policy. MEME performs a MEC between the posterior over the message b^t and distribution over actions $\pi(s^t)$, as determined by the MDP policy. Let $\nu = \text{MEC}(b^t, \pi(s^t))$ denote joint distribution over messages and actions resulting from the coupling. Then MEME sets the sender to act according to the message conditional distribution

$$\pi_{|M}^t(s^t, m) = \nu(A^t | M = m)$$

of the coupling distribution $\nu = \text{MEC}(b^t, \pi(s^t))$, as is described in Algorithm 1.

Algorithm 1 MEME (Sender)

Input: MaxEnt MDP policy π , MCG G_{MCG}
 message $m \leftarrow \text{reset}(G_{\text{MCG}})$ // observe message
 belief $b \leftarrow G_{\text{MCG}}.\mu$
 state $s \leftarrow G_{\text{MCG}}.s^0$
while sender's turn **do**
 joint $\nu \leftarrow \text{minimum_entropy_coupling}(b, \pi(s))$
 decision rule $\pi_{|M}(s) \leftarrow \nu(A | M)$
 sender action $a \sim \pi_{|M}(s, m)$
 new belief $b \leftarrow \text{posterior_update}(b, \pi_{|M}(s), a)$
 next state $s \leftarrow G_{\text{MCG}}.\text{step}(a)$
end while

Given the sender's MDP trajectory, MEME's receiver uses the sender's MDP policy and MEC procedure to reconstruct the sender's message conditional policy along the trajectory; thereafter, the receiver computes the posterior and guesses the maximum a posteriori (MAP) message, as shown in Algorithm 2.

5.2. Method Analysis

MEME possesses guarantees both concerning the return $\mathcal{R}(Z)$ generated by the MDP and concerning the amount of information communicated $I(M; Z)$.

Proposition 1. *At any state, in expectation over messages, the message conditional policy $\pi_{|M}$ selects actions with the same probabilities as the MDP policy π .*

Algorithm 2 MEME (Receiver)

Input: MaxEnt MDP policy π , MCG G_{MCG}
 MDP trajectory $z \leftarrow G_{\text{MCG}}.\text{receiver_observation}()$
 belief $b \leftarrow G_{\text{MCG}}.\mu$
for $s, a \in z$ **do**
 joint $\nu \leftarrow \text{minimum_entropy_coupling}(b, \pi(s))$
 decision rule $\pi_{|M}(s) \leftarrow \nu(A | M)$
 new belief $b \leftarrow \text{posterior_update}(b, \pi_{|M}(s), a)$
end for
 estimated message $\hat{m} \leftarrow \arg \max_{m'} b(m')$
 $G_{\text{MCG}}.\text{step}(\hat{m})$

Proof. Fix an arbitrary state s . Let b be a posterior over the message induced by the sender's message conditional policy on the way to s . Then recall that MEME uses the distribution $\text{MEC}(b, \pi(s))$ to select its action. Because MECs guarantee that the resultant joint distribution marginalizes correctly, it follows directly that MEME must select its actions with the same probabilities as $\pi(s)$. \square

Note that Proposition 1 only holds in expectation over messages. For particular messages, the distribution over actions will generally differ from the MaxEnt policy.

Proposition 2. *In expectation over messages, the expected return generated from the MDP by the message conditional policy $\pi_{|M}$ is equal to that of the MaxEnt policy π . That is,*

$$\mathbb{E} [\mathcal{R}(Z) | \pi_{|M}] = \mathbb{E} [\mathcal{R}(Z) | \pi].$$

Proof. It follows from Proposition 1 that all trajectories are generated with the same probabilities and therefore that the expected returns are the same. \square

Note that, as with Proposition 1, Proposition 2 only holds in expectation over messages. For particular messages, the expected return of the message conditional $\pi_{|M}$ will generally differ from that of the MaxEnt policy π .

Proposition 3. *At each decision point, MEME greedily maximizes the mutual information $I(M; H^{t+1} | b^t, h^t)$ between the message M the history at the next time step H^{t+1} , given the contemporaneous posterior and history, subject to Proposition 1.*

Proof. For conciseness, we leave the conditional dependence on b^t and h^t implicit in the argument. First, we claim that $I(M; H^{t+1}) = I(M; A^t)$. To see this, first consider that $H^{t+1} \equiv (h^t, A^t, S^{t+1})$. This means we have $I(M; H^{t+1}) = I(M; (A^t, S^{t+1}))$ since we are conditioning on h^t . Now, consider that because the message influences the next state only by means of the action, we have the causal graph $M \rightarrow A^t \rightarrow (A^t, S^{t+1})$, which implies that $M \perp (A^t, S^{t+1}) | A^t$. Also, we trivially have

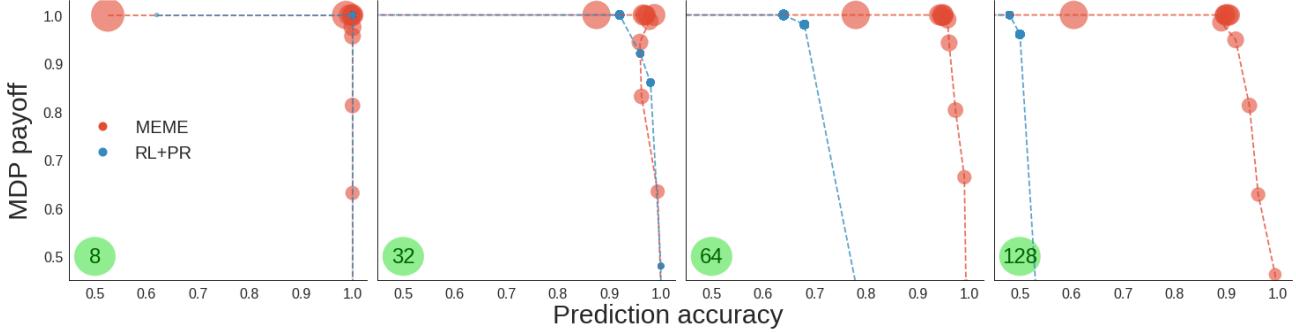


Figure 7. Results for MEME and RL+PR on CodeGrid with varying message space sizes.

$M \perp A^t \mid (A^t, S^{t+1})$. These conditional independence properties allow us to apply the data processing inequality: $X \perp Z \mid Y \Rightarrow I(X; Y) \geq I(X; Z)$. Applying it in both directions yields $I(M; (A^t, S^{t+1})) = I(M; A^t)$.

Now consider that we can rewrite mutual information using the equality

$$I(M; A^t) = H(M) + H(A^t) - H(M, A^t).$$

The first term $H(M)$ is exogenous by virtue of being determined by b^t . The second term is exogenous by virtue of being subject to Proposition 1. The third term is the joint entropy between M and A^t , which is exactly what a minimum entropy coupling minimizes. \square

MaxEnt RL The analysis thus far concerns justifying MEME’s iterative minimum entropy coupling procedure with formal guarantees. While we do not provide guarantees with analogous rigor for the MaxEnt RL component, there is nevertheless strong justification for its usage. Recall that MEME is concerned with i) maximizing the MDP return and ii) greedily maximizing the amount of information communicated at each decision point. These two quantities are $\mathcal{R}(Z)$ and $H(A^t) + H(M) - H(M, A^t)$ where the second quantity corresponds to the one-step mutual information, as was shown in the proof of Proposition 3. Consider the four terms here. The term in black text, $H(M)$, is exogenously specified, and therefore can be ignored for optimization purposes. The term in red text, $-H(M, A^t)$, is exactly the quantity being maximized by minimum entropy coupling. Elegantly, the remaining terms (shown in blue text), are exactly the quantities that MaxEnt RL is concerned with maximizing—the MDP return $\mathcal{R}(Z)$ and the entropy over actions $H(A^t)$.

MaxEnt RL is also advantageous in that it allows MEME to trade off between MDP return and communication using the temperature α . For large temperatures, MEME places more emphasis on maximizing $H(A^t)$ (and thereby $I(M; Z)$),

while for smaller temperatures, MEME places more emphasis on maximizing $\mathcal{R}(Z)$. The appropriate choice of α for a particular setting will depend on the message priority ζ . Code for MEME illustrating the functionality of α applied to the example from Figure 1 can be found at <https://bit.ly/36I3LDm> (the code is executable in browser).

5.3. Scalability

Historically, performing even approximate versions of the minimum entropy couplings described in Algorithms 1 and 2 would have been impossible due to a lack of polynomial minimum entropy coupling methods. However, with the recent advent of efficient approximation algorithms (Cicalese et al., 2019), it is now possible to perform near-optimal couplings in $O(N \log N)$ time. Still, $O(N \log N)$ is not scalable to very large message spaces. To accommodate this fact, for large message spaces, we use a factored representation $\mathbb{M} \subset \mathbb{M}_1 \times \dots \times \mathbb{M}_k$ and a corresponding factored belief (b_1, \dots, b_k) , where each b_j tracks the posterior over \mathbb{M}_j . Specifically, at each time step, we perform a MEC between $\pi(s^t)$ and the block $b_j = \arg \max_{b_j} H(b_j)$ having the largest entropy.¹ By doing so, the size of the space over which the minimum entropy coupling is performed is greatly reduced, allowing us to scale to extremely large message spaces while retaining the guarantees of Proposition 2. While Proposition 3 is no longer satisfied—because i) we use approximate minimum entropy coupling, ii) we only couple a single message block at a time, and iii) the posterior is only approximate if the message prior is not factorable—mutual information remains high in practice, as we will see in the experiments.

¹Pseudocode for this procedure is provided in the appendix.

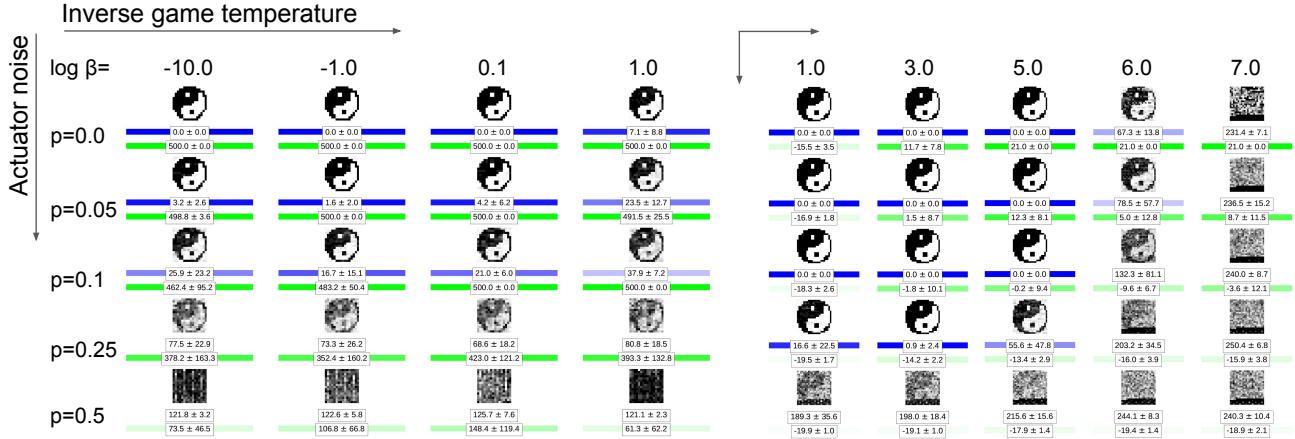


Figure 8. Results for MEME on CodeCart (left) and CodePong (right) with varying amounts of actuator noise and temperatures. The blue bars indicate pixel errors, and the green bars show scores.

6. Experiments

We investigate the efficacy of MEME on MCGs based on a gridworld, Cartpole and Pong. For all three we use MaxEnt Q-learning for our MDP policy. Our codebase is available at <https://github.com/schroederdewitt/meme>.

CodeGrid In our gridworld MCG, which we call CodeGrid, the sender is placed on a 4×4 grid in which it can move left, right, up and down. The sender starts at position $(1, 1)$ and must move to $(4, 4)$ within 8 time steps to achieve the maximal MDP return of 1. Otherwise, the game terminates after 8 time steps and the sender receives a payoff of 0.

For our baseline, we trained an RL agent to play as the MCG sender. We paired this RL agent with a perfect receiver, meaning that, for each episode, the receiver computed the exact posterior over the message based on the sender’s current policy and guessed the MAP, both during training and testing. We abbreviate this baseline as RL+PR (where PR stands for perfect receiver). RL+PR is inspired by related work such as Strouse et al. (2018); Eysenbach et al. (2018). Pseudocode for the RL+PR baseline can be found in the appendix.

We show results for MEME and RL+PR across a variety of settings in Figure 7. The column of the figure corresponds to the cardinality of the message space; the exact size is listed in the green bubble. We use a uniform marginal over messages in each case. The x -axis corresponds to the proportion of the time that the receiver correctly guesses the message. The y -axis corresponds to the MDP payoff (in this case whether the sender reached the opposing corner of the grid within the time limit). Both MEME and RL+PR

possess mechanisms to trade-off between these two goals. MEME can adjust its temperature α , while RL+PR can adjust the value of ζ used during training. Figure 7 show the Pareto curve for each. For MEME, the size of the circle corresponds to the inverse temperature $\beta = 1/\alpha$.

For the settings with 8 messages and 32 messages, we observe that both MEME and RL+PR achieve strong performance—achieving optimal or nearly optimal MDP returns and optimal or nearly optimal transmission. However, as the size of the message space increases, the performance of RL+PR falls off sharply. Indeed, for the 128 message setting, RL+PR is only able to correctly transmit the message slightly more than half the time. RL+PR’s inability to achieve strong performance in these cases may be a result of the fact that communication protocols violate the locality assumption of approximate dynamic programming approaches. In contrast, we observe that MEME, which constructs its protocol using MEC, remains optimal or near optimal both the 64 and the 128 message settings.

CodeCart and CodePong While the CodeGrid experiments show that MEME can outperform an obvious approach to MCGs, it remains to be determined whether MEME can perform well at scale. Toward the end of making this determination, we consider MCGs based on Cartpole and Pong. In these MCGs, the message spaces are the sets of 16×16 and 22×22 binary images, respectively, each with a uniform prior. The cardinality of these spaces ($> 10^{77}$ and $> 10^{145}$) is exceedingly large. In fact, it is not immediately clear how an RL+PR-like approach could even be scaled to this setting. On the other hand, MEME is easily adaptable to this setting, using the factorization scheme suggested in the method section.

We also include results with variable amounts of *actuator noise*, i.e., with some probability p , the environment executes a random action, rather than the one it intended. Actuator noise models the probability of error during transmission and arises naturally in many settings involving communication, such as noisy channel coding. In this setting, the receiver may only observe the action executed by the environment, rather than the one intended by the sender. While this setting is not technically an MCG and Proposition 3 no longer holds (even with perfect MEC), we show MEME to be a strong heuristic in these settings.

We show the results in Figure 8 for CodeCart and CodePong, respectively. For both plots, the y -axis corresponds to the amount of actuator noise (lower is more noise). The x -axis corresponds the inverse temperature value $\beta = 1/\alpha$ (further right is colder temperature, meaning there is more emphasis on MDP expected return). Each entry contains a decoded yin-yang symbol with the corresponding temperature and actuator noise. Each entry also lists the ℓ_1 pixel error (blue) and the MDP expected return (green), along with corresponding standard errors over 10 runs; a brighter color corresponds to better performance.

Remarkably, for both CodeCart and CodePong, we observe that, when there is no actuator noise, MEME is able to perfectly transmit images while achieving the maximum expected return in the MDP. For CodeCart, this occurs at $\log \beta \in \{-10, -1, 0.1\}$; for CodePong, it occurs at $\log \beta \in \{4, 5\}$. Interestingly, as the amount of actuator noise increases to a non-zero value, the effect on performance differs between CodeCart and CodePong. In CodeCart, MEME continues to achieve maximal performance in the MDP, but begins to accumulate errors in the transmission. In contrast, in CodePong, MEME continues to transmit the message with perfect fidelity, but begins to lose expected return from the MDP. This suggests that accidentally taking a random action is costlier in Pong than it is in Cartpole, but that, in an informal sense, Pong has more bandwidth to transmit information. That said, in both cases, the decay in performance is graceful—for example, with $p = 0.05$, the decrease in the visual quality of the transmission for Cartpole ($\log \beta = -1$) is difficult to even perceive, while the CodePong sender still manages to win games roughly 80% of the time ($\log \beta = 5$). The performance in both settings continues to deteriorate up to $p = 1/2$, at which point MEME is neither able to perform adequately on the MDP nor to transmit a clear symbol.

Videos of play are included in the supplemental material.

7. Conclusions and Future Work

This work introduces a new problem setting called Markov coding games, which generalize referential games and

source coding and are related to channel coding problems and decentralized control. We contribute an algorithm for MCGs called MEME and show that MEME possesses provable guarantees regarding both the return generated from the MDP and the amount of information content communicated. Experimentally, we show that MEME significantly outperforms an RL baseline with a perfect receiver. Finally, we show that MEME is able to scale to extremely large message spaces and transmit these messages with high fidelity, even with some actuator noise, suggesting that it could be robust in real world settings.

Security One direction for future work is applying MEME as a method for secure communication. Consider a situation in which an agent would like to communicate privileged information to an allied agent, while under adversarial observation. One possibility is for the agent to communicate using an encrypted channel. However, doing so informs the adversary that the agent possibly has access to, and is communicating, sensitive information. This is undesirable in many contexts, as it may result in the adversary launching attacks in order to determine the sensitive information or prevent it from being communicated. An alternative is for the agent to perform a task as it normally would and allow the ally (the receiver) to interpret the message from the trajectory. This allows the agent to deny that communication ever took place, due to Proposition 1.

Iterative Minimum Entropy Coupling As a subprocedure of MEME, this work made contributions in showing how distributions can be iteratively approximately minimum entropy coupled. Specifically, Step 2 of factored MEME demonstrates how any distribution factorable into blocks with small supports can be iteratively coupled with an autoregressively specified distribution. Another direction for future work is to extend this approach so that it may be applied more generally, with two autoregressively specified distributions (rather than requiring one be factorable). Such an approach could improve MEME’s performance in settings in which the belief prior is not factorable. It could prove useful in more general contexts as a tool for producing low entropy couplings of distributions with large supports.

8. Acknowledgements

We thank Frans A. Oliehoek, Alexander Robey, and Yiding Jiang for helpful discussions. This work was supported by the Bosch Center for Artificial Intelligence; the Cooperative AI Foundation; the Cyber Defence Campus, Science and Technology, Armasuisse, Switzerland; the Microsoft Research PhD Scholarship Program; the Microsoft Research EMEA PhD Award; the European Research Council under the European Union’s Horizon 2020 research and innovation programme (grant agreement #637713); and NVIDIA.

References

- Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, Jun 2013. ISSN 1076-9757. doi: 10.1613/jair.3912. URL <http://dx.doi.org/10.1613/jair.3912>.
- Bhatt, V. and Buro, M. Inference-based deterministic messaging for multi-agent communication. *ArXiv*, abs/2103.02150, 2021.
- Chen, J. and Berger, T. The capacity of finite-state markov channels with feedback. *IEEE Transactions on Information Theory*, 51(3):780–798, 2005. doi: 10.1109/TIT.2004.842697.
- Cicalese, F., Gargano, L., and Vaccaro, U. Minimum-entropy couplings and their applications. *IEEE Transactions on Information Theory*, 65(6):3436–3451, 2019. doi: 10.1109/TIT.2019.2894519.
- Das, A., Gervet, T., Romoff, J., Batra, D., Parikh, D., Rabbat, M., and Pineau, J. Tarmac: Targeted multi-agent communication. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1538–1546. PMLR, 2019.
- Eysenbach, B., Gupta, A., Ibarz, J., and Levine, S. Diversity is all you need: Learning skills without a reward function. *CoRR*, abs/1802.06070, 2018. URL <http://arxiv.org/abs/1802.06070>.
- Farrell, J. and Rabin, M. Cheap talk. *Journal of Economic Perspectives*, 10(3):103–118, September 1996. doi: 10.1257/jep.10.3.103. URL <https://www.aeaweb.org/articles?id=10.1257/jep.10.3.103>.
- Foerster, J., Assael, I. A., de Freitas, N., and Whiteson, S. Learning to communicate with deep multi-agent reinforcement learning. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 29, pp. 2137–2145. Curran Associates, Inc., 2016. URL <https://proceedings.neurips.cc/paper/2016/file/c7635bfd99248a2cdef8249ef7bfbef4-Paper.pdf>.
- Golin, M. J., Kenyon, C., and Young, N. E. Huffman coding with unequal letter costs. In *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing*, STOC ’02, pp. 785–791, New York, NY, USA, 2002. Association for Computing Machinery. ISBN 1581134959. doi: 10.1145/509907.510020. URL <https://doi.org/10.1145/509907.510020>.
- Iwata, K.-i., Morii, M., and Uyematsu, T. An efficient universal coding algorithm for noiseless channel with symbols of unequal cost. *IEICE Transactions on Fundamentals*, pp. 2232–2237, 01 1997.
- Kostina, V. and Verdú, S. Fixed-length lossy compression in the finite blocklength regime: Gaussian source. In *2011 IEEE Information Theory Workshop*, pp. 457–461, 2011. doi: 10.1109/ITW.2011.6089501.
- Kovačević, M., Stanojević, I., and Šenk, V. On the entropy of couplings. *Information and Computation*, 242:369–382, 2015. ISSN 0890-5401. doi: <https://doi.org/10.1016/j.ic.2015.04.003>. URL <https://www.sciencedirect.com/science/article/pii/S0890540115000450>.
- Larmore, L. L. and Hirschberg, D. S. Length-limited coding. In *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA ’90, pp. 310–318, USA, 1990. Society for Industrial and Applied Mathematics. ISBN 0898712513.
- Lathi, B. P. *Modern Digital and Analog Communication Systems 3e Osece*. Oxford University Press, Inc., USA, 3rd edition, 1998. ISBN 0195110099.
- Lazaridou, A., Hermann, K. M., Tuyls, K., and Clark, S. Emergence of Linguistic Communication from Referential Games with Symbolic and Pixel Input. *arXiv:1804.03984 [cs]*, April 2018. URL <http://arxiv.org/abs/1804.03984>. arXiv: 1804.03984.
- Lewis, D. K. *Convention: A Philosophical Study*. Wiley-Blackwell, 1969.
- MacKay, D. J. C. *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, USA, 2002. ISBN 0521642981.
- Mao, H., Zhang, Z., Xiao, Z., Gong, Z., and Ni, Y. Learning multi-agent communication with double attentional deep reinforcement learning. *Auton. Agents Multi Agent Syst.*, 34(1):32, 2020a. URL <https://doi.org/10.1007/s10458-020-09455-w>.
- Mao, H., Zhang, Z., Xiao, Z., Gong, Z., and Ni, Y. Learning agent communication under limited bandwidth by message pruning. In *AAAI*, 2020b.
- Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. Human-level control through deep reinforcement learning. *Nature*, 518 (7540):529–533, February 2015. ISSN 1476-4687. doi: <https://doi.org/10.1038/nature14236>.

- 10.1038/nature14236. URL <https://www.nature.com/articles/nature14236>. Number: 7540 Publisher: Nature Publishing Group.
- Nair, V. and Hinton, G. E. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pp. 807–814, Madison, WI, USA, June 2010. Omnipress. ISBN 978-1-60558-907-7.
- Oliehoek, F. A., Amato, C., et al. *A concise introduction to decentralized POMDPs*, volume 1. Springer, 2016.
- Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., Devito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in PyTorch, 2017.
- Skyrms, B. *Signals: Evolution, Learning, and Information*. Oxford University Press, 2010.
- Smith, K. The cultural evolution of communication in a population of neural networks. *Connection Science*, 14, 04 2002. doi: 10.1080/09540090210164306.
- Spike, M., Stadler, K., Kirby, S., and Smith, K. Minimal requirements for the emergence of learned signaling. *Cognitive Science*, 41(3):623–658, 2017. doi: <https://doi.org/10.1111/cogs.12351>. URL <https://onlinelibrary.wiley.com/doi/abs/10.1111/cogs.12351>.
- Steels, L. Self-organization and selection in cultural language evolution. *Experiments in Cultural Language Evolution*, 03 2013. doi: 10.1075/ais.3.02ste.
- Strouse, D. J., Kleiman-Weiner, M., Tenenbaum, J., Botvinick, M., and Schwab, D. Learning to share and hide intentions using information regularization. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pp. 10270–10281, Red Hook, NY, USA, 2018. Curran Associates Inc.
- Sukhbaatar, S., szlam, a., and Fergus, R. Learning multiagent communication with backpropagation. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL <https://proceedings.neurips.cc/paper/2016/file/55b1927fdafef39c48e5b73b5d61ea60-Paper.pdf>.
- Sutton, R. S. and Barto, A. G. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018. ISBN 978-0-262-03924-6.
- Tatikonda, S. and Mitter, S. The capacity of channels with feedback. *IEEE Transactions on Information Theory*, 55(1):323–349, 2009. doi: 10.1109/TIT.2008.2008147.
- Wang, H. S. and Moayeri, N. Finite-state markov channel-a useful model for radio communication channels. *IEEE Transactions on Vehicular Technology*, 44(1):163–171, 1995. doi: 10.1109/25.350282.
- Wang, R., He, X., Yu, R., Qiu, W., An, B., and Rabinovich, Z. Learning efficient multi-agent communication: An information bottleneck approach. In *ICML*, pp. 9908–9918, 2020. URL <http://proceedings.mlr.press/v119/wang20i.html>.
- Ziebart, B. D., Maas, A., Bagnell, J. A., and Dey, A. K. Maximum entropy inverse reinforcement learning. In *Proceedings of the 23rd national conference on Artificial intelligence - Volume 3*, AAAI'08, pp. 1433–1438, Chicago, Illinois, July 2008. AAAI Press. ISBN 978-1-57735-368-3.

A. Experimental Details

For CodeGrid, we use a policy parameterized by neural network with two fully-connected layers of hidden dimension 64, each followed by a ReLu activation (Nair & Hinton, 2010). For CodePong and CodeCart, we use a convolutional encoder with three layers of convolutions (number of channels, kernel size, stride) as follows: (32,8,4), (64,4,2), (64,3,1). This is followed by a fully connected layer (Mnih et al., 2015). We use ReLu activations after each layer, note that we do not use any max-pooling. For CodeGrid and CodePong, layer weights are randomly initialized using PyTorch 1.7 (Paszke et al., 2017) defaults. We used 200k training episodes for CodeGrid and 2M training episodes for CodePong. For CodeCart, we initialized weights according to an optimally-trained DQN policy included in *rl-baselines3-zoo*² and individually fine-tune for each β for 15k steps. For all environments, we used the Adam optimizer with learning rate 10^{-4} , $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$ and no weight decay. The results in Figures 7 and 8 are averages over 10 rollouts; for Figure 8, uncertainty is shown with standard error.

CodeGrid

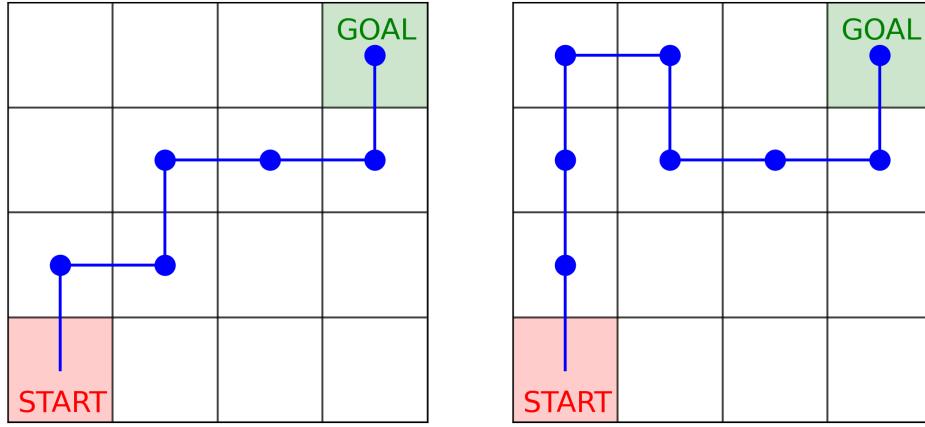


Figure 9. An illustration of two possible CodeGrid trajectories, one of length 6 (left) and one of length 8 (right).

CodeCart For CodeCart we use the standard action space (move left and move right). There are no redundant actions that the agent can use to communicate.

CodePong For CodePong we use a reduced action space of size two (move up and move down). There are no redundant actions that the agent can use to communicate.

²<https://github.com/DLR-RM/rl-baselines3-zoo>

B. Algorithmic Details

B.1. Factored MEME

Algorithm 3 Factored MEME

// Step 1: Compute MDP policy
Input: MDP G_{MDP} , temperature α
 $\text{MaxEnt MDP policy } \pi \leftarrow \text{MaxEntRL}(G_{MDP}, \alpha)$

// Step 2: Play Sender's Part of MCG episode

Input: MDP policy π , MCG G_{MCG}
message $m \leftarrow \text{reset}(G_{MCG})$ // observe message from environment
belief $b \leftarrow G_{MCG}.\mu$
state $s \leftarrow G_{MCG}.s^0$
while sender's turn **do**
 active block index $i \leftarrow \arg \max_j \{\mathcal{H}(b_j) | b_j \in b\}$
 joint $\nu \leftarrow \text{minimum_entropy_coupling}(b_i, \pi(s))$
 decision rule $\pi_{|M} \leftarrow \nu(A | M)$
 sender action $a \sim \pi_{|M}(m_i)$
 new belief $b_i \leftarrow \text{posterior_update}(b_i, \pi_{|M}, a)$
 next state $s \leftarrow G_{MCG}.\text{step}(a)$
end while

// Step 3: Play Receiver's Part of MCG episode

Input: MDP policy π , MCG G_{MCG}
MDP trajectory $z \leftarrow G_{MCG}.\text{receiver_observation}()$
belief $b \leftarrow G_{MCG}.\mu$
for $s, a \in z$ **do**
 active block index $i \leftarrow \arg \max_j \{\mathcal{H}(b_j) | b_j \in b\}$
 joint $\nu \leftarrow \text{minimum_entropy_coupling}(b_i, \pi(s))$
 decision rule $\pi_{|M} \leftarrow \nu(A | M)$
 new belief $b \leftarrow \text{posterior_update}(b, \pi_{|M}, a)$
end for
for i **do**
 estimated i th message block $\hat{m}_i \leftarrow \arg \max_{m'_i} b_i(m'_i)$
end for
 $G_{MCG}.\text{step}(\hat{m})$

B.2. RL+PR baseline

Algorithm 4 RL+PR baseline

```

// Play Episode
Input: MCG  $G_{\text{MCG}}$ 
state  $s \leftarrow s^0$ 
message  $m \leftarrow \text{sample}(\mu)$ 
belief  $b \leftarrow \mu$ 
while True do
    action  $a \leftarrow \text{sample}(\pi(s, m))$ 
    belief  $b \leftarrow \text{posterior\_update}(b, \pi, a)$ 
    new state  $s' \leftarrow \text{sample}(\mathcal{T}(s, a))$ 
    if  $s'$  non-terminal then
        add_to_buffer( $s, a, \mathcal{R}(s, a), s'$ )
        state  $s \leftarrow s'$ 
    else
        break
    end if
end while
add_to_buffer( $s, a, \mathcal{R}(s, a) + \zeta \max_{m' \in \mathcal{M}} b(m'), \emptyset$ )
  
```

B.3. Min Entropy Joint Distribution Algorithm outputting a sparse representation of M (Cicalese et al., 2019)
Algorithm 5 Min Entropy Joint Distribution

Require: prob. distributions $\mathbf{p} = (p_1, \dots, p_n)$ and $\mathbf{q} = (q_1, \dots, q_n)$
Ensure: A Coupling $\mathbf{M} = [m_{ij}]$ of \mathbf{p} and \mathbf{q} in sparse representation $\mathbf{L} = \{(m_{ij}, (i, j)) \mid m_{ij} \neq 0\}$

```

if  $\mathbf{p} \neq \mathbf{q}$ , let  $i = \max \{j \mid p_j \neq q_j\}$ ; if  $p_i < q_i$  then swap  $\mathbf{p} \leftrightarrow \mathbf{q}$ .
 $\mathbf{z} = (z_1, \dots, z_n) \leftarrow \mathbf{p} \wedge \mathbf{q}$ ,  $\mathbf{L} \leftarrow \emptyset$ 
CreatePriorityQueue( $\mathcal{Q}^{(row)}$ ),  $qrowsum \leftarrow 0$ 
CreatePriorityQueue( $\mathcal{Q}^{(col)}$ ),  $qcolsum \leftarrow 0$ 
for  $i = n$  downto 1 do
     $z_i^{(d)} \leftarrow z_i$ ,  $z_i^{(r)} \leftarrow 0$ 
    if  $qcolsum + z_i > q_i$  then
         $(z_i^{(d)}, z_i^{(r)}, I, qcolsum) \leftarrow \text{Lemma3-Sparse}(z_i, q_i, \mathcal{Q}^{(col)}, qcolsum)$ 
    else
        while  $\mathcal{Q}^{(col)} \neq \emptyset$  do
             $(m, l) \leftarrow \text{ExtractMin}(\mathcal{Q}^{(col)})$ ,
             $qcolsum \leftarrow qcolsum - m$ ,
             $\mathbf{L} \leftarrow \mathbf{L} \cup \{(m, (l, i))\}$ 
        end while
    end if
    if  $qrowsum + z_i > p_i$  then
         $(z_i^{(d)}, z_i^{(r)}, I, qrowsum) \leftarrow \text{Lemma3-Sparse}(z_i, p_i, \mathcal{Q}^{(row)}, qrowsum)$ 
        for each  $(m, l) \in I$  do  $\mathbf{L} \rightarrow \mathbf{L} \cup \{(m, (i, l))\}$ 
        if  $z_i^{(r)} > 0$  then Insert  $(\mathcal{Q}^{(row)}, (z_i^{(r)}, i))$ 
         $qrowsum \leftarrow qrowsum + z_i^{(r)}$ 
    else
        while  $\mathcal{Q}^{(row)} \neq \emptyset$  do
             $(m, l) \leftarrow \text{ExtractMin}(\mathcal{Q}^{(row)})$ 
             $qrowsum \leftarrow qrowsum - m$ 
             $\mathbf{L} \leftarrow \mathbf{L} \cup \{(m, (i, l))\}$ 
        end while
    end if
     $\mathbf{L} \leftarrow \mathbf{L} \cup \{(z_i^{(d)}, (i, i))\}$ 
end for

```

Algorithm 6 Lemma3-Sparse

Require: real $z > 0$, $x \geq 0$, and priority queue \mathcal{Q} s.t. $\left(\sum_{(m,l) \in \mathcal{Q}} m\right) = qsum$ and $qsum = x \geq z$
Ensure: $z^{(d)}, z^{(r)} \geq 0$, and $I \subseteq \mathcal{Q}$ s.t. $z^{(d)} + z^{(r)} = z$, and $z^{(d)} + \sum_{(m,l) \in I} m = x$

```

 $I \leftarrow \emptyset$ ,  $sum \leftarrow 0$ 
while  $\mathcal{Q} \neq \emptyset$  and  $sum + \text{Min}(\mathcal{Q}) < x$  do
     $(m, l) \leftarrow \text{ExtractMin}(\mathcal{Q})$ ,  $qsum \leftarrow qsum - m$ 
     $I \leftarrow I \cup \{(m, l)\}$ ,  $z^{(r)} \leftarrow z - z^{(d)}$ 
end while
 $z^{(d)} \leftarrow x - sum$ ,  $z^{(r)} \leftarrow z - z^{(d)}$ 
return  $(z^{(d)}, z^{(r)}, I, qsum)$ 

```