

# Learning to Ground Multi-Agent Communication with Autoencoders

**Toru Lin**  
MIT CSAIL  
torulk@mit.edu

**Minyoung Huh**  
MIT CSAIL  
minhuh@mit.edu

**Chris Stauffer**  
Facebook AI  
cstauffer@fb.com

**Ser-Nam Lim**  
Facebook AI  
sernamlim@fb.com

**Phillip Isola**  
MIT CSAIL  
phillipi@mit.edu

## Abstract

Communication requires having a common language, a lingua franca, between agents. This language could emerge via a consensus process, but it may require many generations of trial and error. Alternatively, the lingua franca can be given by the environment, where agents ground their language in representations of the observed world. We demonstrate a simple way to ground language in learned representations, which facilitates decentralized multi-agent communication and coordination. We find that a standard representation learning algorithm – autoencoding – is sufficient for arriving at a grounded common language. When agents broadcast these representations, they learn to understand and respond to each other’s utterances and achieve surprisingly strong task performance across a variety of multi-agent communication environments.

## 1 Introduction

An essential aspect of communication is that each pair of speaker and listener must share a common understanding of the symbols being used [9]. For artificial agents interacting in an environment, with a communication channel but without an agreed-upon communication protocol, this raises the question: *how can meaningful communication emerge before a common language has been established?*

To address this challenge, prior works have used supervised learning [19], centralized learning [15, 18, 30], or differentiable communication [7, 18, 30, 34, 43]. Yet, none of these mechanisms is representative of how communication emerges in nature, where animals and humans have evolved communication protocols without supervision and without a centralized coordinator [37]. The communication model that most closely resembles language learning in nature is a fully decentralized model, where agents’ policies are independently optimized. However, decentralized models perform poorly even in simple communication tasks [18] or with additional inductive biases [11].

We tackle this challenge by first making the following observations on why emergent communication is difficult in a decentralized multi-agent reinforcement learning setting. A key problem that prevents agents from learning meaningful communication is the lack of a common grounding in communication symbols [3, 11, 20]. In nature, the emergence of a common language is thought to be aided by physical biases and embodiment [31, 44] – we can only produce certain vocalizations, these sounds only can be heard a certain distance away, these sounds bear similarity to natural sounds in the environment, etc. – yet artificial communication protocols are not a priori grounded in aspects of the environment

Project page, code, and videos can be found at <https://toruowo.github.io/marl-ae-comm/>.

dynamics. This poses a severe exploration problem as the chances of a consistent protocol being found and rewarded is extremely small [18]. Moreover, before a communication protocol is found, the random utterances transmitted between agents add to the already high variance of multi-agent reinforcement learning, making the learning problem even more challenging [11, 30].

To overcome the grounding problem, an important question to ask is: do agents really need to learn language grounding from scratch through random exploration in an environment where success is determined by chance? Perhaps nature has a different answer; previous studies in cognitive science and evolutionary linguistics [21, 38, 41, 42] have provided evidence for the hypothesis that communication first started from sounds whose meaning are grounded in the physical environment, then creatures adapted to make sense of those sounds and make use of them. Inspired by language learning in natural species, we propose a novel framework for grounding multi-agent communication: first ground *speaking* through learned representations of the world, then learn *listening* to interpret these grounded utterances. Surprisingly, even with the simple representation learning task of autoencoding, our approach eases the learning of communication in fully decentralized multi-agent settings and greatly improves agents’ performance in multi-agent coordination tasks that are nearly unsolvable without communication.

The contribution of our work can be summarized as follows:

- We formulate communication grounding as a representation learning problem and propose to use observation autoencoding to learn a common grounding across all agents.
- We experimentally validate that this is an effective approach for learning decentralized communication in MARL settings: a communication model trained with a simple autoencoder can consistently outperform baselines across various MARL environments.
- In turn, our work highlights the need to rethink the problem of emergent communication, where we demonstrate the essential need for visual grounding.

## 2 Related Work

In multi-agent reinforcement learning (MARL), achieving successful emergent communication with decentralized training and non-differentiable communication channel is an important yet challenging task that has not been satisfactorily addressed by existing works. Due to the non-stationary and non-Markovian transition dynamics in multi-agent settings, straightforward implementation of standard reinforcement learning methods such as Actor-Critic [24] and DQN [33] perform poorly [18, 30].

Centralized learning is often used to alleviate the problem of high variance in MARL, for example learning a centralized value function that has access to the joint observation of all agents [10, 15, 30]. However, it turns out that MARL models are unable to solve tasks that rely on emergent communication, even with centralized learning and shared policy parameters across the agents [18]. Eccles et al. [11] provides an analysis that illustrates how MARL with communication poses a more difficult exploration problem than standard MARL, which is confirmed by empirical results in [18, 30]: communication exacerbates the sparse and high variance reward signal in MARL.

Many works therefore resort to differentiable communication [7, 18, 30, 34, 43], where agents are allowed to directly optimize each other’s communication policies through gradients. Among them, Choi et al. [7] explore a high-level idea that is similar to ours: generating messages that the model itself can interpret. However, these approaches impose a strong constraint on the nature of communication, which limits their applicability to many real-world multi-agent coordination tasks.

Jaques et al. [22] proposes a method that allows independently trained RL agents to communicate and coordinate. However, the proposed method requires that an agent either has access to policies of other agents or stays in close proximity to other agents. These constraints make it difficult for the same method to be applied to a wider range of tasks, such as those in which agents are not embodied or do not observe others directly. Eccles et al. [11] attempts to solve the same issue by introducing inductive biases for positive signaling and positive listening, but implementation requires numerous task-specific hyperparameter tuning, and the effectiveness is limited.

It is also worth noting that, while a large number of existing works on multi-agent communication take structured state information as input [5, 16, 17, 18, 34, 36], we train agents to learn a communication protocol directly from raw pixel observations. This presents additional challenges due to

the unstructured and ungrounded nature of pixel data, as shown in [3, 7, 26]. To our knowledge, this work is the first to effectively use representation learning to aid communication learning from pixel inputs in a wide range of MARL task settings.

### 3 Preliminaries

We model **multi-agent reinforcement learning (MARL) with communication** as a partially-observable general-sum Markov game [28, 40], where each agent can broadcast information to a shared communication channel. Each agent receives a partial observation of the underlying world state at every time step, including all information communicated in the shared channel. This observation is used to learn an appropriate policy that maximizes the agent’s environment reward. In this work, we parameterize the policy function using a deep neural network.

Formally, a decentralized MARL can be expressed as a partially observable Markov decision process as  $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{C}, \mathcal{O}, \mathcal{T}, R, \gamma \rangle$ , where  $N$  is the number of agents,  $\mathcal{S}$  is a set of states spaces,  $\mathcal{A} = \{\mathcal{A}^1, \dots, \mathcal{A}^N\}$ ,  $\mathcal{C} = \{\mathcal{C}^1, \dots, \mathcal{C}^N\}$ , and  $\mathcal{O} = \{\mathcal{O}^1, \dots, \mathcal{O}^N\}$  are a set of action, of communication, and of observation spaces respectively.

At time step  $t$ , an agent  $k$  observes a partial view  $o_t^{(k)}$  of the underlying true state  $s_t$ , and a set of communicated messages from the previous time step  $c_{t-1} = \{c_{t-1}^{(1)}, \dots, c_{t-1}^{(N)}\}$ . The agent then chooses an action  $a_t^{(k)} \in \mathcal{A}^k$  and a subsequent message to broadcast  $c_t^{(k)} \in \mathcal{C}^k$ . Given the joint actions of all  $N$  agents  $a_t = \{a_t^{(1)}, \dots, a_t^{(N)}\} \in (\mathcal{A}^1, \dots, \mathcal{A}^N)$ , the transition function  $\mathcal{T} : \mathcal{S} \times \mathcal{A}^1 \times \dots \times \mathcal{A}^N \rightarrow \Delta(\mathcal{S})$  maps the current state  $s_t$  and set of agent actions  $a_t$  to a distribution over the next state  $s_{t+1}$ . Since the transition function  $\mathcal{T}$  is non-deterministic, we denote the probability distributions over  $\mathcal{S}$  as  $\Delta(\mathcal{S})$ . Finally, each agent receives an individual reward  $r_t^{(k)} \in R(s_t, a_t)$  where  $R : \mathcal{S} \times \mathcal{A}^1 \times \dots \times \mathcal{A}^N \rightarrow \mathbb{R}$ .

In our work, we consider a fully cooperative setting in which the objective of each agent is to maximize the total expected return of all agents:

$$\underset{\pi: \mathcal{S} \rightarrow \mathcal{A} \times \mathcal{C}}{\text{maximize}} \quad \mathbb{E} \left[ \sum_{t \in T} \sum_{k \in N} \gamma^t R(s_t, a_t) \mid (a_t, c_t) \sim \pi^{(k)}, s_t \sim \mathcal{T}(s_{t-1}) \right] \quad (1)$$

for some finite time horizon  $T$  and discount factor  $\gamma$ .

In MARL, the aforementioned objective function is optimized using policy gradient. Specifically, we use asynchronous advantage actor-critic (A3C) [32] with Generalized Advantage Estimation [39] to optimize our policy network. The policy network in A3C outputs a distribution over the actions and the discounted future returns. Any other policy optimization algorithms could have been used in lieu of A3C. We do not use centralized training or self-play, and only consider decentralized training where each agent is parameterized by an independent policy.

### 4 Grounding Representation for Communication with Autoencoders

The main challenge of learning to communicate in fully decentralized MARL settings is that there is no grounded information to which agents can associate their symbolic utterances. This lack of grounding creates a dissonance across agents and poses a difficult exploration problem. Ultimately, the gradient signals received by agents are therefore largely inconsistent. As the time horizon, communication space, and the number of agents grow, this grounding problem becomes even more pronounced. This difficulty is highlighted in numerous prior works, with empirical results showing that agents often fail to use the communication channel at all during decentralized learning [3, 11, 20, 30].

We propose a simple and surprisingly effective approach to mitigate this issue: using a self-supervised representation learning task to learn a common grounding [45] across all agents. Having such a grounding enables speakers to communicate messages that are understandable to the listeners and convey meaningful information about entities in the environment, though agents need not use the same symbols to mean the same things. Specifically, we train each agent to independently learn to auto-encode its own observation and use the learned representation for communication. This approach offers the benefits of allowing fully decentralized training without needing additional architectural

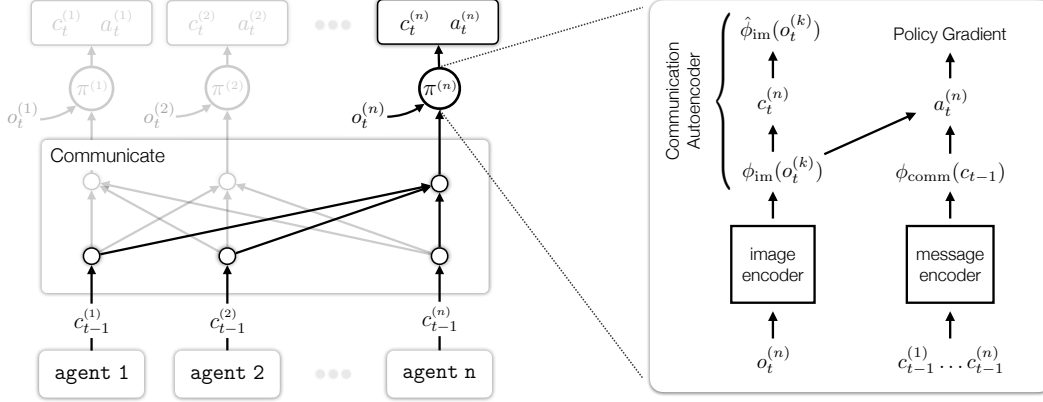


Figure 1: **Overview:** The overall schematic of our multi-agent system. All agents share the same individual model architecture, but each agent is independently trained to learn to auto-encode its own observation and use the learned representation for communication. At each time step, each agent observes an image representation of the environment as well as messages broadcasted by other agents during the last time step. The image pixels are processed through an Image Encoder; the broadcasted messages are processed through a Message Encoder; the image features and the message features are concatenated and passed through a Policy Network to predict the next action. The image features are also used to generate the next communication messages using the Communication Autoencoder.

bias or supervision signal. In Section 5, we show the effectiveness of our approach on a variety of MARL communication tasks.

An overview of our method is shown in Figure 1, which illustrates the communication flow among agents at some arbitrary time step  $t$ . All agents share the same individual model architecture, and each agent consists of two modules: a speaker module and a listener module. We describe the architecture details of a single agent  $k$  below.

#### 4.1 Speaker Module

At each time step  $t$ , the speaker module takes in the agent’s observation  $o_t^{(k)}$  and outputs the agent’s next communication message  $c_t^{(k)}$ .

**Image Encoder** Given the raw pixel observation, the module first uses an image encoder to embed the pixels into a low-dimensional feature  $o_t^{(k)} \rightarrow \phi_{im}(o_t^{(k)}) \in \mathbb{R}^{128}$ . The image encoder is a convolutional neural network with 4 convolutional layers, and the output of this network is spatially pooled. We use the same image encoder in the listener module.

**Communication Autoencoder** The goal of the communication autoencoder is to take the current state observation and generate the next subsequent message. We use an autoencoder to learn a mapping from the feature space of image encoder to communication symbols, i.e.  $\phi_{im}(o_t^{(k)}) \rightarrow c_t^{(k)}$ . The autoencoder consists of an encoder and a decoder, both parameterized by a 3-layer MLP. The decoder tries to reconstruct the input state from the communication message  $c_t^{(k)} \rightarrow \hat{\phi}_{im}(o_t^{(k)})$ . The communication messages are quantized before being passed through the decoder. We use a straight-through estimator to differentiate through the quantization [1]. The auxiliary objective function of our model is to minimize the reconstruction loss  $\|\phi_{im}(o_t^{(k)}) - \hat{\phi}_{im}(o_t^{(k)})\|_2^2$ . This loss is optimized jointly with the policy gradient loss from the listener module.

#### 4.2 Listener Module

While the goal of the speaker module is to output grounded communication based on the agent’s private observation  $o_t^{(k)}$ , the goal of the listener module is to learn an optimal action policy based on both the observation  $o_t^{(k)}$  and communicated messages  $c_{t-1}$ . At each time step  $t$ , the listener module outputs the agent’s next action  $a_t^{(k)}$ .

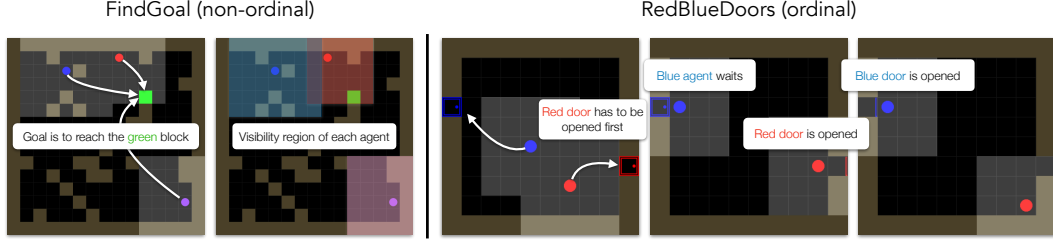


Figure 2: **MarlGrid Environment:** We introduce two new grid environments: FindGoal (left) and RedBlueDoors (right). These environments are adapted from the GridWorld environment [6, 35]. Environment states are randomized at every episode and are partially observable to the agents. In FindGoal, the task is to reach the green goal location. Each agent receives a reward of 1 when they reach the goal, and an additional reward of 1 when all 3 agents reach the goal within the time frame. In RedBlueDoors, the task is ordinal, where the ordering of actions matter. A reward of 1 is given to both agents if and only if the red door is opened first and then the blue door.

**Message Encoder** The message encoder linearly projects all messages communicated from the previous time step  $c_{t-1}$  using a shared embedding layer. The information across all agent message embeddings is combined through concatenation and passed through 3-layer MLP. The resulting message feature has a fixed dimension of 128, i.e.  $\phi_{\text{comm}}(c_t) \in \mathbb{R}^{128}$ .

**Policy Network** Each agent uses an independent policy head, which is a standard GRU [8] policy with a linear layer. The GRU policy concatenates the encoded image features and the message features  $\phi_t^{(k)} = \phi_{\text{im}}(o_t^{(k)}) \circ \phi_{\text{comm}}(c_t)$ , where  $\circ$  is the concatenation operator across the feature dimension. The GRU policy predicts a distribution over the actions  $a \sim \pi(\phi_t^{(k)})$  and the corresponding expected returns. The predicted action distribution and expected returns are used for computing the policy gradient loss. This loss is jointly optimized with the autoencoder reconstruction loss from the speaker module.

The same setup is used for all experiments. The exact details of the network architecture and the corresponding training details are in the Appendix.

## 5 Experiments

In this section, we demonstrate that autoencoding is a simple and effective representation learning algorithm to ground communication in MARL. We evaluate our method on various multi-agent environments and qualitatively show that our method outperforms baseline methods. We then provide further ablations and analyses on the learned communication.

### 5.1 Environments

We introduce three multi-agent communication environments: CIFAR Game, FindGoal, and RedBlueDoors. Our work focuses on fully cooperative scenarios, but can also be extended to competitive or mixed scenarios.

Our environments cover a wide range of communication task settings, including (1) referential or non-referential, (2) ordinal or non-ordinal, and (3) two-agent versus generalized multi-agent. A referential game, often credited to Lewis signaling game [27], refers to a setup in which agents communicate through a series of message exchanges to solve a task. In contrast to non-referential games, constructing a communication protocol is critical to solving the task – where one can only arrive at a solution through communication. Referential games are referred to as a grounded learning environment, and therefore, communication in MARL has been studied mainly through the lens of referential games [13, 26]. Lastly, ordinal tasks refer to a family of problems where the ordering of the actions is critical for solving the task. The difference between ordinal and non-ordinal settings is illustrated in Figure 2, where the blue agent must wait for the red agent to open the door to successfully receive a reward. In contrast, non-ordinal tasks could benefit from shared information, but it is not necessary to complete the task. We now describe the environments used in our work in more detail:

**CIFAR Game** We design CIFAR Game following the setup of Multi-Step MNIST Game in [18], but with CIFAR-10 dataset [25] instead. This is a non-ordinal, two-agent, referential game. In CIFAR game, each agent independently observes a randomly drawn image from the CIFAR-10 dataset, and the goal is to communicate the observed image to the other agent within 5 environment time steps. At each time step, each agent broadcasts a set of communication symbols of length  $l$ . At the final time step, each agent must choose a class label from the 10 possible choices. At the end of the episode, an agent receives a reward of 0.5 for each correctly guessed class label, and both agents receive a reward of 1 only when both images are classified correctly.

**MarlGrid Environments** The second and third environments we consider are: FindGoal (Figure 2 left) and RedBlueDoors (Figure 2 right). Both environments are adapted from the GridWorld environment [6, 35] and environment states are randomized at every episode.

FindGoal is a non-ordinal, multi-agent, non-referential game. We use  $N = 3$  agents, and at each time step, each agent observes a partial view of the environment centered at its current position. The task of agents is to reach the green goal location as fast as possible. Each agent receives an individual reward of 1 for completing the task and an additional reward of 1 when all agents have reached the goal. Hence, the optimal strategy of an agent is to communicate the goal location once it observes the goal. If all agents learn a sufficiently optimized search algorithm, they can maximize their reward without communication.

RedBlueDoors is an ordinal, two-agent, non-referential game. The environment consists of a red door and a blue door, both initially closed. The task of agents is to open both doors, but unlike in the previous two games, the ordering of actions executed by agents matters. A reward of 1 is given to both agents if and only if the red door is opened first and then the blue door. This means that any time the blue door is opened first, both agents receive a reward of 0, and the episode ends immediately. Hence, the optimal strategy for agents is to convey the information that the red door was opened. Since it is possible to solve the task through visual observation or by a single agent that opens both doors, communication is not necessary.

Compared to CIFAR Game, the MarlGrid environments have a higher-dimensional observation space and a more complex action space. The fact that these environments are non-referential exacerbates the visual-language grounding problem since communication can only exist in the form of *cheap talk* (i.e., costless, nonbinding, nonverifiable communication [14] that has no direct effect on the game state and agent payoffs). We hope to show from this set of environments that autoencoders can be used as a surprisingly simple and adaptable representation learning task to ground communication. It requires little effort to implement and almost no change across environments. Most importantly, as we will see in Section 5.4, autoencoded representation shows an impressive improvement over communication trained with reinforcement learning.

## 5.2 Baselines

To evaluate the effectiveness of grounded communication, we compare our method (ae-comm) against the following baselines: (1) a no-comm baseline, where agents are trained without a communication channel; (2) a rl-comm baseline\*, where we do not make a distinction between the listener module and the speaker module, and the communication policy is learned in a way similar to the environment policy; (3) a rl-comm-with-biases baseline, where inductive biases for positive signaling and positive listening are added to rl-comm training as proposed in [11]; (4) a ae-rl-comm baseline, where the communication policy is learned by an additional policy network trained on top of the autoencoded representation in speaker module.

## 5.3 The Effectiveness of Grounded Communication

In Table 1 and Figure 3, we compare task performance of ae-comm agents with performance of baseline agents. We report all results with 95% confidence intervals, evaluated on 100 episodes per seed over 10 random seeds.

In CIFAR Game environment, all three baselines (no-comm, rl-comm, rl-comm-with-biases) could only obtain an average reward close to that of random guesses throughout the training process.

\*The rl-comm baseline can be seen as an A3C version of RIAL [18] without parameter sharing.



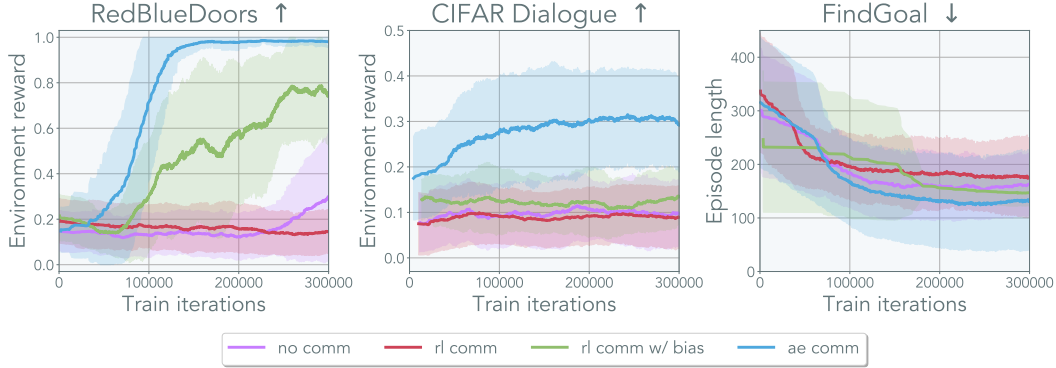


Figure 3: **Comparison with baselines:** Comparison between our method that uses an autoencoded communication (ae-comm), a baseline that is trained without communication (no-comm), a baseline where communication policy is trained using reinforcement learning (rl-comm), and another baseline where inductive biases for positive signaling and positive listening are added to rl-comm training (rl-comm-with-inductive-biases). For FindGoal, we visualize the amount of time it takes for all agents to reach the goal, as all methods can reach the goal within the time frame. For each set of results, we report the mean and 95% confidence intervals evaluated on 100 episodes per seed over 10 random seeds.

In comparison, ae-comm agents achieve a much higher reward on average. Since this environment is a referential game, our results directly indicate that ae-comm agents learn to communicate more effectively than the baseline agents.

RedBlueDoors environment poses a challenging multi-agent coordination problem since the reward is extremely sparse. As shown in Figure 3(b), neither of the no-comm and rl-comm agents was able to learn a successful coordination strategy. Although rl-comm-with-biases agents outperform the other two baseline agents, they do not learn an optimal strategy that guarantees an average reward close to 1. In contrast, ae-comm agents converge to an optimal strategy after 150k of training.

methods	CIFAR	RedBlueDoors	FindGoal
	avg. $r \uparrow$	avg. $r \uparrow$	avg. $t \downarrow$
no-comm	$0.082 \pm 0.009$	$0.123 \pm 0.096$	$169.0 \pm 26.8$
rl-comm	$0.099 \pm 0.013$	$0.174 \pm 0.009$	$184.8 \pm 25.2$
rl-comm w/ bias [11]	$0.142 \pm 0.019$	$0.729 \pm 0.072$	$158.0 \pm 12.3$
ae-comm (ours)	<b><math>0.348 \pm 0.041</math></b>	<b><math>0.984 \pm 0.002</math></b>	<b><math>103.5 \pm 20.2</math></b>

Table 1: **Comparison with baselines:** We compute the average reward for CIFAR Game and RedBlueDoors environments, and average episode length for FindGoal environment. For each set of results, we report the mean and 95% confidence intervals evaluated on 100 episodes per seed over 10 random seeds.

In FindGoal environment, agents are able to solve the task without communication, but their performance can be improved with communication. Therefore, we use episode length instead of reward as the performance metric for this environment. To resolve the ambiguity in Figure 3(c), we also include numerical results in a table below. While all agents are able to obtain full rewards, Figure 3 shows that ae-comm agents are able to complete the episode much faster than other agents. We further verify that this improvement is indeed a result of the successful communication by providing further analysis in Section 5.5.

Our results indicate that a communication model trained with autoencoding tasks consistently outperforms the baselines across all environments. The observation that communication does not work well with reinforcement learning is consistent with observations made in prior works [11, 18, 30]. Furthermore, our results with autoencoders – a task that is often considered trivial – highlight that we as a community may have overlooked a critical representation learning component in MARL communication. In Section 5.4, we provide a more detailed discussion on the difficulty of training emergent communication via policy optimization.

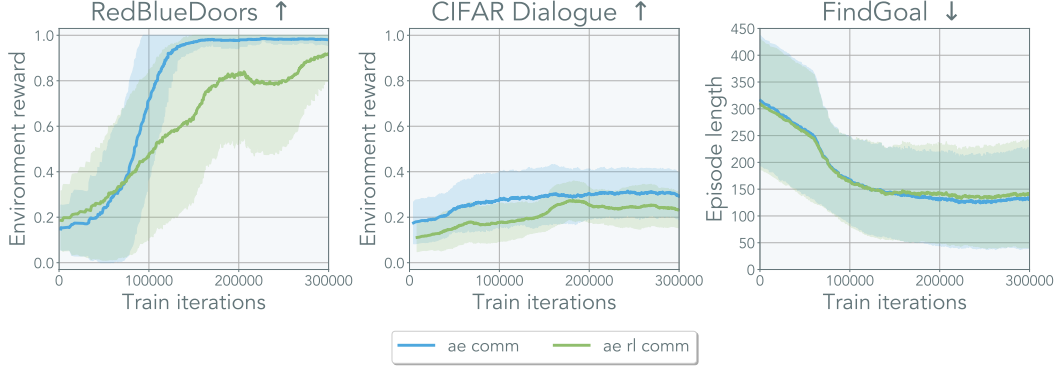


Figure 4: **Representation learning with reinforcement learning:** Comparison between a speaker module trained with only an autoencoding task (ae-comm) and another one trained with both autoencoding task and reinforcement learning (ae-rl-comm). We observed that further training a policy on top of the autoencoder representation degrades performance across all environments (except for in FindGoal, where performance of ae-rl-comm and ae-comm stayed about the same).

#### 5.4 The Role of Autoencoding

Given the success of agents trained with autoencoders, it is natural to ask whether a better communication protocol can emerge from the speaker module by jointly training it with a reinforcement learning policy. To this end, we train a GRU policy on top of the autoencoded representation (ae-rl-comm) and compare it against our previous model that was trained just with an autoencoder (ae-comm). The communication policy head is independent of the environment action policy head.

Surprisingly, we observed in Figure 4 that the model trained jointly with reinforcement learning consistently performed worse (except for in FindGoal, where performance of ae-rl-comm and ae-comm stayed about the same). We hypothesize that the lack of correlation between visual observation and the communication rewards hurts the agents’ performance. This lack of visual-reward grounding could introduce conflicting gradient updates to the action policy, and thereby exacerbate the high-variance problem that already exists in reinforcement learning. In other words, optimization is harder whenever the joint-exploration problem for learning speaker and listener policies is introduced. Our observation suggests that specialized reward design and training at the level of [2] might be required for decentralized MARL communication. This prompts us to rethink how to address the lack of visual grounding in communication policy, where this work serves as a first step in this direction.

#### 5.5 Analyzing the Effects of Communication Signals on Agent Behavior

**Communication Embedding** To analyze whether the agents have learned a meaningful visual grounding for communication, we first visualize the communication embedding. In Figure 5, we visualize the communication symbols transmitted by the agents trained on RedBlueDoors. The communication symbols are discrete with a length of  $l = 10$  (1024 possible embedding choices), and we use approximately 4096 communication samples across 10 episodic runs. We embed the communication symbols using t-SNE [46] and further cluster them using DBSCAN [12]. In the figure, we visualize clusters by observing the correspondence between image states and communication symbols produced by agents in those states. For example, we observed that a specific communication cluster corresponded to an environment state when the red door was opened; this suggests a communication action where one agent signals the other agent to open the blue door and complete the task.

**Entropy of Action Distribution** To measure whether communicated information directly influences other agents’ actions, we visualize the entropy of action distribution during episode rollouts. Suppose one agent shares information that is vital to solving the task. In that case, a decrease in entropy should be observed in the action distributions of other agents, as they act more deterministically towards solving the task.

As shown in Figure 6, we visualize the entropy of action distribution across 256 random episodic runs using policy parameters from a fully trained ae-comm model. The entropies are aligned using



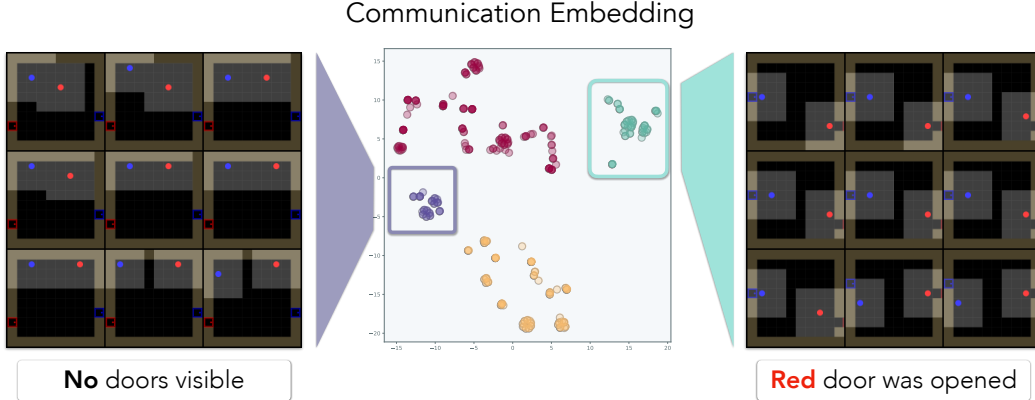


Figure 5: **Communication clusters:** 4096 communication messages are embedded into low-dimensional representation using t-SNE [46] and is clustered using DBSCAN [12]. We visualize the images corresponding to the communication messages. We observed that the message clusters correspond to various meaningful phases throughout the task. The communication symbol of the purple cluster corresponds to when no doors are visible by either agent, and the light green cluster corresponds to when the red door is opened.

environment milestone events: for FindGoal, this is when the first agent reaches the goal; for RedBlueDoors, this is when the red door is opened. Since the identity of the agents that solve the task first does not matter, entropy plots are computed with respect to the *listener* agents (i.e., agents that receive vital information from others). In FindGoal, this corresponds to the last agent to reach the goal; in RedBlueDoors, this corresponds to the agent opening the blue door. For both environments, we see a sharp fall-off in entropy as soon as the first agents finish the task. In contrast, agents trained without autoencoding act randomly regardless of whether other agents have completed the task. This reaffirms that the agents trained with an autoencoder can effectively transmit information to other agents.

#### Positive Signaling and Positive Listening

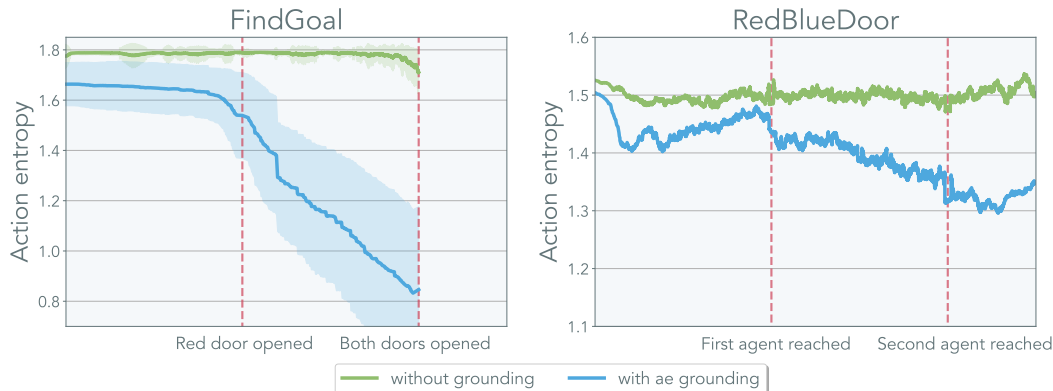
We additionally investigate the two metrics suggested by [29] for measuring effectiveness of communication, *positive signaling* and *positive listening*. Since ae-comm agents have to communicate their learned representation, the presence of representation learning task loss means that ae-comm agents are intrinsically optimized for *positive signaling* (i.e., sending messages that are related to their observation or action). In Table 2, we report the increase in average reward when adding a communication channel, comparing ae-comm with its baselines; this metric is suggested by [29] to be a sufficient metric for *positive listening* (i.e., communication messages influence the behavior of agents). We observe that agent task performance improves most substantially in ae-comm with the addition of communication channel.

methods	CIFAR	RedBlueDoors
	gain $\Delta r \uparrow$	gain $\Delta r \uparrow$
rl-comm	$0.017 \pm 0.022$	$-0.030 \pm 0.155$
rl-comm w/ bias [11]	$0.060 \pm 0.029$	$0.552 \pm 0.155$
ae-comm (ours)	<b><math>0.266 \pm 0.058</math></b>	<b><math>0.807 \pm 0.185</math></b>

Table 2: **Performance gain with communication:** Positive listening as measured by the increase in reward after adding a communication channel. Performance of ae-comm agents improves more than the baselines.

## 6 Discussion and Societal Impacts

We present a framework for grounding multi-agent communication through autoencoding, a simple self-supervised representation learning task. Our method allows agents to learn non-differentiable communication in fully decentralized settings and does not impose constraints on input structures (e.g., state inputs or pixel inputs) or task nature (e.g., referential or non-referential). Our results demonstrate that agents trained with the proposed method achieve much better performance on a suite of coordination tasks compared to baselines.



**Figure 6: Policy entropy with communication:** We visualize the entropy of the action policy throughout the task (lower is better). The graph is generated with 256 random episodes. For FindGoal, the entropy is measured on the last agent to enter the goal, and for RedBlueDoors the entropy is measured on the agent that opens the blue door (the second door). All 256 runs are aligned to the *dotted red lines* which corresponds to the time in which the first and second agent enters the goal for (FindGoal), and the time in which the red and the blue doors are opened for (RedBlueDoors). The model trained with an auto-encoder transmits messages that are effectively used by other agents.

We believe this work on multi-agent communication is of importance to our society for two reasons. First, it extends a computational framework under which scientific inquiries concerning language acquisition, language evolution, and social learning can be made. Second, unlike works in which agents can only learn latent representations of other agents through passive observations [47, 48], it opens up new ways for artificial learning agents to improve their coordination and cooperative skills, increasing their reliability and usability when deployed to real-world tasks and interacting with humans.

However, we highlight two constraints of our work. First, our method assumes that all agents have the same model architecture and the same autoencoding loss, while real-world applications may involve heterogeneous agents. Second, our method may fail in scenarios where agents need to be more selective of the information communicated since they are designed to communicate their observed information indiscriminately. We believe that testing the limits of these constraints, and relaxing them, will be important steps for future work.

We would also like to clarify that communication between the agents in our work is only part of what true “communication” should eventually entail. We position our work as one partial step toward true communication, in that our method provides a strong bias toward positive signaling and allows decentralized agents to coordinate their behavior toward common goals by passing messages amongst each other. It does not address other aspects of communication, such as joint optimization between a speaker and a listener. In the appendix section, we highlight the difficulty of training emergent communication via policy optimization alongside empirical results.

Another limitation of this work, which is also one concern we have regarding its potential negative societal impact, is that the environments we consider are fully cooperative. If the communication method we present in this work is to be deployed to the real world, we need to either make sure the environment is rid of adversaries, or conduct additional research to come up with robust counter-strategies in the face of adversaries, which could use better communication policies as a way to lie, spread misinformation, or maliciously manipulate other agents.

## Acknowledgements

We sincerely thank all the anonymous reviewers for their extensive discussions on and valuable contributions to this paper. We thank Lucy Chai and Xiang Fu for helpful comments on the manuscript. We thank Jakob Foerster for providing inspiring advice on multi-agent training. Additionally, TL would like to thank Sophie and Sofia; MH would like to thank Sally, Leo and Mila; PI would like to thank Moxie and Momo. This work was supported by a grant from Facebook.

## References

- [1] Y. Bengio, N. Léonard, and A. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. 4, 14
- [2] C. Berner, G. Brockman, B. Chan, V. Cheung, P. D  biak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019. 8
- [3] B. Bogin, M. Geva, and J. Berant. Emergence of communication in an interactive world with consistent speakers. *arXiv preprint arXiv:1809.00549*, 2018. 1, 3
- [4] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016. 13
- [5] K. Cao, A. Lazaridou, M. Lanctot, J. Z. Leibo, K. Tuyls, and S. Clark. Emergent communication through negotiation. *arXiv preprint arXiv:1804.03980*, 2018. 2
- [6] M. Chevalier-Boisvert, L. Willems, and S. Pal. Minimalistic gridworld environment for openai gym. <https://github.com/maximecb/gym-minigrid>, 2018. 5, 6
- [7] E. Choi, A. Lazaridou, and N. de Freitas. Compositional oververter communication learning from raw visual input. *arXiv preprint arXiv:1804.02341*, 2018. 1, 2, 3
- [8] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014. 5
- [9] A. Dafoe, E. Hughes, Y. Bachrach, T. Collins, K. R. McKee, J. Z. Leibo, K. Larson, and T. Graepel. Open problems in cooperative ai. *arXiv preprint arXiv:2012.08630*, 2020. 1
- [10] A. Das, T. Gervet, J. Romoff, D. Batra, D. Parikh, M. Rabbat, and J. Pineau. Tarmac: Targeted multi-agent communication. In *International Conference on Machine Learning*, pages 1538–1546. PMLR, 2019. 2
- [11] T. Eccles, Y. Bachrach, G. Lever, A. Lazaridou, and T. Graepel. Biases for emergent communication in multi-agent reinforcement learning. *arXiv preprint arXiv:1912.05676*, 2019. 1, 2, 3, 6, 7, 9
- [12] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996. 8, 9
- [13] K. Evtimova, A. Drozdov, D. Kiela, and K. Cho. Emergent communication in a multi-modal, multi-step referential game. *arXiv preprint arXiv:1705.10369*, 2017. 5
- [14] J. Farrell. Cheap talk, coordination, and entry. *The RAND Journal of Economics*, pages 34–39, 1987. 6
- [15] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. 1, 2
- [16] J. Foerster, F. Song, E. Hughes, N. Burch, I. Dunning, S. Whiteson, M. Botvinick, and M. Bowling. Bayesian action decoder for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 1942–1951. PMLR, 2019. 2
- [17] J. N. Foerster, Y. M. Assael, N. de Freitas, and S. Whiteson. Learning to communicate to solve riddles with deep distributed recurrent q-networks. *arXiv preprint arXiv:1602.02672*, 2016. 2
- [18] J. N. Foerster, Y. M. Assael, N. De Freitas, and S. Whiteson. Learning to communicate with deep multi-agent reinforcement learning. *arXiv preprint arXiv:1605.06676*, 2016. 1, 2, 6, 7
- [19] L. Graesser, K. Cho, and D. Kiela. Emergent linguistic phenomena in multi-agent communication games. *arXiv preprint arXiv:1901.08706*, 2019. 1
- [20] S. Harnad. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1-3):335–346, 1990. 1, 3
- [21] J. R. Hurford. Biological evolution of the saussurean sign as a component of the language acquisition device. *Lingua*, 77(2):187–222, 1989. 2
- [22] N. Jaques, A. Lazaridou, E. Hughes, C. Gulcehre, P. Ortega, D. Strouse, J. Z. Leibo, and N. De Freitas. Social influence as intrinsic motivation for multi-agent deep reinforcement learning. In *International Conference on Machine Learning*, pages 3040–3049. PMLR, 2019. 2
- [23] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 13
- [24] V. R. Konda and J. N. Tsitsiklis. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014. Citeseer, 2000. 2
- [25] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. (0), 2009. 6
- [26] A. Lazaridou, K. M. Hermann, K. Tuyls, and S. Clark. Emergence of linguistic communication from referential games with symbolic and pixel input. *arXiv preprint arXiv:1804.03984*, 2018. 3, 5
- [27] D. Lewis. *Convention: A philosophical study*. John Wiley & Sons, 2008. 5
- [28] M. L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pages 157–163. Elsevier, 1994. 3
- [29] R. Lowe, J. Foerster, Y.-L. Boureau, J. Pineau, and Y. Dauphin. On the pitfalls of measuring emergent communication. *arXiv preprint arXiv:1903.05168*, 2019. 9
- [30] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *arXiv preprint arXiv:1706.02275*, 2017. 1, 2, 3, 7

- [31] B. MacWhinney. *The emergence of language from embodiment*. Psychology Press, 2013. 1
- [32] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016. 3
- [33] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013. 2
- [34] I. Mordatch and P. Abbeel. Emergence of grounded compositional language in multi-agent populations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. 1, 2
- [35] K. Ndousse. marlgrid. <https://github.com/kandouss/marlgrid>, 2020. 5, 6
- [36] M. Noukhovitch, T. LaCroix, A. Lazaridou, and A. Courville. Emergent communication under competition. *arXiv preprint arXiv:2101.10276*, 2021. 2
- [37] M. A. Nowak and D. C. Krakauer. The evolution of language. *Proceedings of the National Academy of Sciences*, 96(14):8028–8033, 1999. 1
- [38] D. K. Roy and A. P. Pentland. Learning words from sights and sounds: A computational model. *Cognitive science*, 26(1):113–146, 2002. 2
- [39] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015. 3
- [40] L. S. Shapley. Stochastic games. *Proceedings of the national academy of sciences*, 39(10):1095–1100, 1953. 3
- [41] L. Steels. The synthetic modeling of language origins. *Evolution of communication*, 1(1):1–34, 1997. 2
- [42] L. Steels and F. Kaplan. Aibo’s first words: The social learning of language and meaning. *Evolution of communication*, 4(1):3–32, 2000. 2
- [43] S. Sukhbaatar, A. Szlam, and R. Fergus. Learning multiagent communication with backpropagation. *arXiv preprint arXiv:1605.07736*, 2016. 1, 2
- [44] M. Tomasello. *Origins of human communication*. MIT press, 2010. 1
- [45] T. Udagawa and A. Aizawa. A natural language corpus of common grounding under continuous and partially-observable context. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7120–7127, 2019. 3
- [46] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 8, 9
- [47] A. Xie, D. P. Losey, R. Tolsma, C. Finn, and D. Sadigh. Learning latent representations to influence multi-agent interaction. *arXiv preprint arXiv:2011.06619*, 2020. 10
- [48] L. Zintgraf, S. Devlin, K. Ciosek, S. Whiteson, and K. Hofmann. Deep interactive bayesian reinforcement learning via meta-learning. *arXiv preprint arXiv:2101.03864*, 2021. 10

## A Implementation Details

### A.1 Environment

All environments used in this work were implemented using OpenAI Gym [4]. Below, we describe the MarlGrid Environments in more detail.

**FindGoal** The game map is a  $15 \times 15$  grid world. On each episode reset, 1 goal tile and 25 obstacle tiles are randomly placed on the map. Then, 3 agents are randomly placed on the remaining empty space. Each agent can only observe a  $7 \times 7$  partial view of the map centered on the agent. Each agent has 5 actions: up, right, down, left, stay. The maximum episode length allowed is 512 time steps.

**RedBlueDoors** The game map is a  $10 \times 10$  grid world. On each episode reset, 1 blue door and 1 red door are placed at a random position on either the leftmost side or the rightmost side of the map; the doors must be on opposite sides. Then, 2 agents are randomly placed on the remaining empty space. Each agent can only observe a  $3 \times 3$  partial view of the map centered on the agent. Each agent has 6 actions: up, right, down, left, stay, open door. The maximum episode length allowed is 2048 time steps.

### A.2 Model

**Image Encoder** The Image Encoder is a convolutional neural network with 4 convolutional layers. Each layer has a kernel size of 3, stride of 2, padding of 1, and outputs 32 channels. ELU activation is applied to each convolutional layer. A 2D adaptive average pooling is applied over the output from convolutional layers. The final output has 32 channels, a height of 3, and a width of 3.

**Communication Autoencoder** The Communication Autoencoder takes as input the output from the Image Encoder. The encoder is a 3-layer MLP with hidden units [128, 64, 32] and ReLU activation. The decoder is a 3-layer MLP with hidden units [32, 64, 128] and ReLU activation. The output communication message is a 1D vector of length 10.

**Message Encoder** The Message Encoder first projects all input messages using an embedding layer of size 32, then concatenates and passes the message embeddings through a 3-layer MLP with hidden units [32, 64, 128] and ReLU activation. Dimension of the output message feature is 128.

**Policy Network** Each policy network is consisted of a GRU policy with hidden size 128, a linear layer mapping GRU outputs to policy logits for the environment action, and a linear layer mapping GRU outputs to the baseline value function.

## B Training Details

### B.1 Hyperparameters

For all experiments, we use the Adam optimizer [23] with a learning rate of 0.0001 and parameters  $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$ . We perform a sweep over the following values of these hyperparameter to select the combination that gives rise to best results: (0.01, 0.001, 0.0001, 0.00001) for learning rate, (0.9, 0.99, 0.995) for  $\beta_1$ , (0.995, 0.999) for  $\beta_2$ , and  $(10^{-7}, 10^{-8})$  for  $\epsilon$ .

### B.2 Compute Resources

We ran all experiments on an internal cluster that consists of 4 NVIDIA GeForce RTX 2080 Ti GPUs.

methods	CIFAR	RedBlueDoors	FindGoal
	avg. $r \uparrow$	avg. $r \uparrow$	avg. $t \downarrow$
no-comm	$0.082 \pm 0.009$	$0.123 \pm 0.096$	$169.0 \pm 26.8$
fc-comm	$0.107 \pm 0.021$	$0.382 \pm 0.022$	$188.3 \pm 33.6$
latent-comm-1	$0.129 \pm 0.034$	$0.714 \pm 0.073$	$160.0 \pm 13.0$
latent-comm-2	$0.092 \pm 0.012$	$0.739 \pm 0.080$	$228.6 \pm 21.3$
full-latent-comm	$0.346 \pm 0.070$	$0.912 \pm 0.046$	$111.1 \pm 26.1$
ae-comm (ours)	<b><math>0.348 \pm 0.041</math></b>	<b><math>0.984 \pm 0.002</math></b>	<b><math>103.5 \pm 20.2</math></b>

Table 3: **Comparison with additional baselines.** We compute the average reward for CIFAR Game and RedBlueDoors environments, and average episode length for FindGoal environment. For each set of results, we report the mean and 95% confidence intervals evaluated on 100 episodes per seed over 10 random seeds.

## C Additional Experiments

### C.1 fc-comm and latent-comm Baselines

In addition to baselines presented in Section 5.2, we added two more baselines: `fc-comm` and `latent-comm`. Below, we describe them in more details, and report comparisons between these additional baselines and our method (`ae-comm`) on all three environments in Table 3.

For `fc-comm`, agents transmit discrete states that take the same form as the autoencoded messages in `ae-comm`. These discrete states come from the policy network without autoencoding. For fair comparison, a fully connected layer is added on top of the policy network states, such that the learned messages for `fc-comm` are discretized and have the same shape and range as messages in `ae-comm`.

For `latent-comm`, agents transmit latent states from before the policy head as communication vectors. For fair comparison, for `latent-comm` we again restrict agents to use the same fixed-size discrete communication channel (except for `full-latent-comm`, which we will explain in the next paragraph). The latent activation vectors in the policy nets are high-dimensional; reducing them to the size of the communication channel can be achieved in multiple ways. In `latent-comm-1`, we do so by reducing the size of the last hidden layer of the policy net so that it matches the size of the communication channel. In `latent-comm-2`, we instead use the original policy architecture and only transmit the first  $N$  features of the last hidden layer of the policy net, where  $N$  is the size of the communication channel. In all cases, we quantize the outputs identically and use a straight-through estimator [1] to differentiate through the quantization. Their performance is better than `fc-comm` baseline; since these latent states are continuously trained with the policy reward, we hypothesize that they contain more useful and relevant information.

As an “upper bound”, we also compare all methods to `full-latent-comm`, a method that communicates the full, high-dimensional and continuous latent vector from the last hidden layer of the policy net. As shown in the table, the performance of `full-latent-comm` is close to that of `ae-comm`. Clearly, if agents can directly observe each other’s full internal states, they can coordinate their behavior well. Our contribution is to show that similar performance can be achieved even when agents are only allowed to make low-dimensional and discrete utterances – a setting that more closely models communication in nature and may have greater practical utility (e.g., low-bandwidth communication channels between robots).

### C.2 Training Emergent Communication via Policy Optimization

Making policy optimization work on top of autoencoding is a difficult yet essential problem to solve. We hypothesize that optimization is harder whenever the joint-exploration problem for learning speaker and listener policies is introduced. We do not have a solution yet, but our empirical results can confirm that the solution is not as simple as requiring more training iterations. We have run training to up to 1 million iterations and did not see an improvement in the result. We have also tried training the autoencoding component and RL component alternatingly, i.e. freezing one while



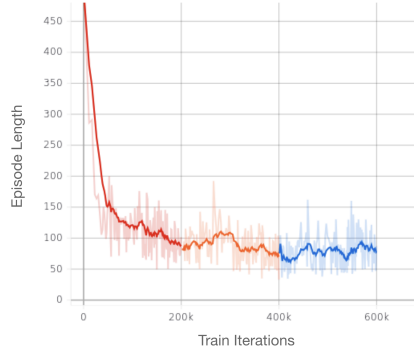


Figure 7: **Difficulty of training policy optimization:** We tried training the autoencoding component and RL component alternately, i.e. freezing one while training the other until loss stabilizes. The alternating training is visualized as different segments in this training graph, starting from the segment trained with autoencoding objective. Performance did not improve after extended training.

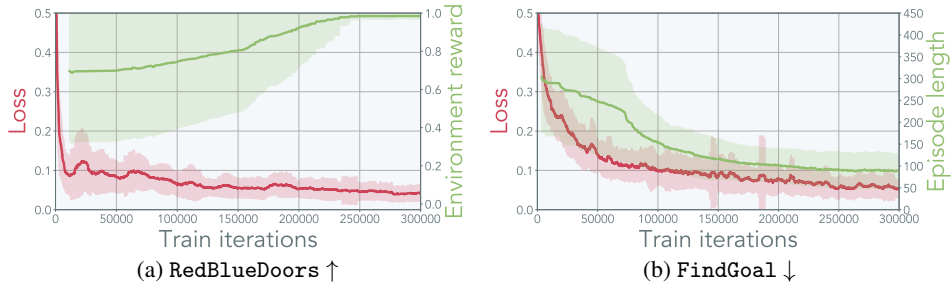


Figure 8: **Performance and representation:** Both agent performance and autoencoder loss improve over the course of learning.

training the other until loss stabilizes, yet performance does not improve whenever the RL objective is optimized. As an example, we include a failed training graph in Figure 7.

### C.3 Autoencoder Loss

In Figure 8, we observe that agent task performance improves as representation learning task loss decreases. In particular, the agent task performance does not improve drastically until representation task loss is sufficiently low.