Learning to cooperate: Emergent communication in multi-agent navigation

Ivana Kajić[1] **(ivana.kajic@uwaterloo.ca)**
Centre for Theoretical Neuroscience, University of Waterloo
Waterloo, ON, Canada

Eser Aygün (eser@google.com)
Doina Precup (doinap@google.com)
DeepMind, Montréal, QC, Canada

Abstract

Emergent communication in artificial agents has been studied to understand language evolution, as well as to develop artificial systems that learn to communicate with humans. We show that agents performing a cooperative navigation task in various gridworld environments learn an interpretable communication protocol that enables them to efficiently, and in many cases, optimally, solve the task. An analysis of the agents' policies reveals that emergent signals spatially cluster the state space, with signals referring to specific locations and spatial directions such as *left*, *up*, or *upper left room*. Using populations of agents, we show that the emergent protocol has basic compositional structure, thus exhibiting a core property of natural language.

Keywords: reinforcement learning; emergent communication; multiagent; cooperative game

Introduction

Natural language use is a form of joint action that requires continuous coordination among the participants sharing a basis for common ground (Clark, 1996). Humans are able to coordinate such actions adaptively, rapidly developing novel communication systems that exhibit core features of natural language such as referential signaling and compositional structure (Bohn, Kachel, & Tomasello, 2019).

Language emergence has also been studied in artificial intelligence using agents with rudimentary communication capabilities, as a means to develop communication systems with characteristics similar to natural language (Bard et al., 2020; Li & Bowling, 2019; Lazaridou, Hermann, Tuyls, & Clark, 2018; Kottur, Moura, Lee, & Batra, 2017). Multi-agent scenarios that use reinforcement learning to train the communicating agents are a promising approach for developing agents that can reason about novel signals while being situated in environments, in contrast to supervised learning methods, which rely on large amounts of static text data (Kottur et al., 2017; Lazaridou et al., 2018).

We expand on this body of research by showing that agents coordinating their actions in a navigation task develop a communication protocol that depends on spatial features of the environment. A population of such agents, implemented with minimal inductive biases, learns a structured communication protocol in which the meaning of the whole signal depends on its constituents, demonstrating the efficiency of learned communication.

Emergent communication is often studied in the context of cooperative games, such as the signaling game of Lewis (1969), where two agents, a sender and a receiver, act jointly to achieve a common goal (Li & Bowling, 2019; Lazaridou et al., 2018). In such a game, the sender sees an artifact (e.g., an image) and uses a communication channel shared with a receiver to transmit a message (e.g., a symbol) from a fixed vocabulary. The receiver then conditions its decision on the message to select the target object among distractors. Both agents are rewarded if the receiver selects the target.

Communication protocols in such games exhibit some level of structure reminiscent of natural language. In particular, compositional structure is found with end-to-end training on disentangled input data (Lazaridou et al., 2018) or by introducing environmental pressures during training (Li & Bowling, 2019).

While referential games provide a useful paradigm for investigating conditions that give rise to communication protocols with natural language-like properties (Kottur et al., 2017), the resulting communication policies are often difficult to interpret (Lowe, Foerster, Boureau, Pineau, & Dauphin, 2019). Moreover, the agents' actions in such setups often do not affect the environment, as the sender and receiver are each restricted to one choice of action.

In this paper, we extend this paradigm by introducing a control component to the task. Specifically, the agent receiving the message needs to interact with the environment in order to navigate to a goal state. We show that the agents are able to develop a communication protocol that is optimal for this task, as demonstrated by their performance, and we perform a qualitative analysis of the protocol. Expanding the setup to allow for multiple senders yields basic compositional structure, where individual messages encode different aspects of the environment, and successful navigation can be achieved solely with action coordination.

The Navigation Task

We consider gridworld navigation tasks of various configurations. In each task, there are two types of agents involved: senders and receivers. In a given task setup, there is exactly one receiver, and at most five senders. At the beginning of each episode ($t = 0$), the goal location is determined by placing a reward at a random, non-occupied location in the gridworld environment. The sender observes the goal location

[1] Research done during an internship at DeepMind.

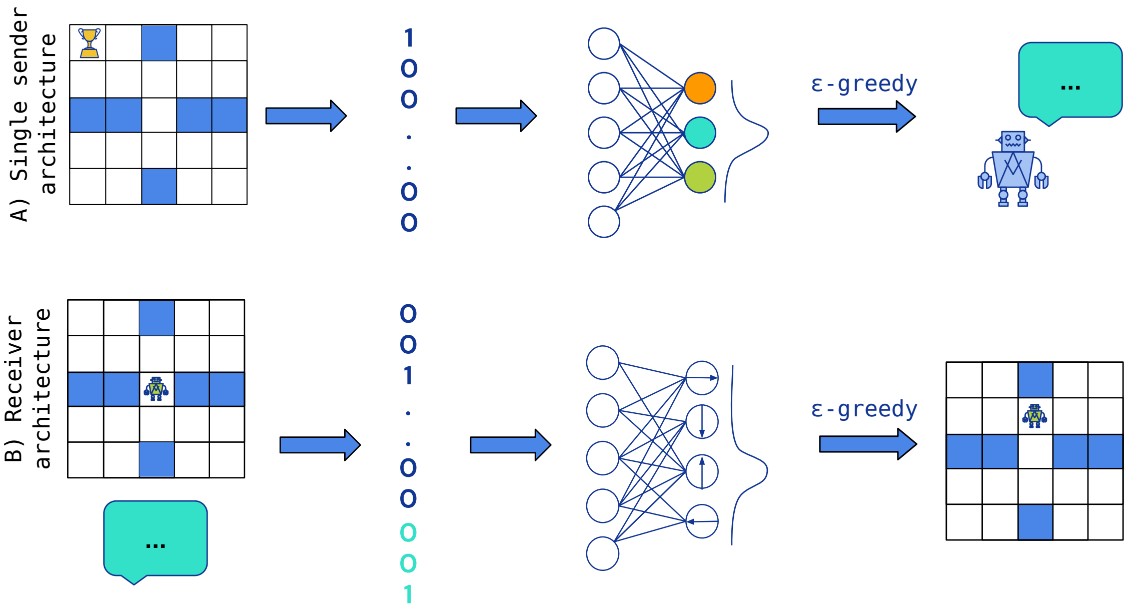

Figure 1: Both the sender and the receiver see the gridworld environment, yet only the sender sees the goal location. It selects a message action (a single symbol) based on the one-hot encoding of the goal location. The receiver selects a navigation action based on the multi-hot input vector that encodes its own location and the message.

and in response, it emits a message m from a vocabulary V, with $|V| = N$. All messages are represented as single discrete symbols (here, we use natural numbers as symbols). The message is stored by the environment and provided as part of the observation available to the receiver. The message observation persists at each subsequent time step $t > 0$. The sender performs no more actions until the end of the episode.

After $t = 0$, the receiver observes its own location (center of the map) and the message, and performs a navigation action. This process repeats until either the goal state is reached or the episode terminates, with probability p_{term}. If the episode terminates because the receiver reached the goal state, both receiver and sender are rewarded with $r = 1$, otherwise $r = 0$.

If the setup involves a population of senders, the interaction remains as described, except that the message emitted at $t = 0$ consists of a sequence of symbols $[m_1, m_2, ..., m_M]$, where m_i is the message emitted by the i-th sender and M is the number of senders. A sender selects its own message action independently of the messages selected by other senders.

Experimental setup

Sender A sender is modelled as a contextual N-armed bandit that selects one message action[2] m out of N possible messages, based on the static context vector $c \in \mathbb{R}^d$, where d is the size (height × weight) of the gridworld.[3]

The context is a one-hot vector that encodes the goal location in the flattened array representing the gridworld. In the experiments, we varied N to study the effects of more compression on task performance.

The i-th sender's action-value estimation function $Q(\cdot)$ is implemented as a single layer feed-forward neural network parameterized by θ_{s_i}. The loss for a single sender is $\mathcal{L}_{s_i} = \left(R_t - Q(c, m_i; \theta_{s_i})\right)^2$ when $t = T$, and 0 when $0 \leq t < T$, where R_t is the reward received at the end of an episode of length T, c is the context and m_i is a message action. Message actions are selected using an ε-greedy policy, where ε is the same for all senders and is determined empirically using hyperparameter search. The implementation of a sender agent is schematically shown in Figure 1A.

Receiver The receiver is implemented as a Q-learning agent (Watkins & Dayan, 1992) with a neural network representing action-values and parameterized by θ_r. After $t = 0$, the environment provides the receiver with an observation $o_t = [p_t, \bar{m}_1, \bar{m}_2, ...]$ at each time step t, where p_t is a one-hot encoding of the receiver position, and each $\bar{m}_i$ is a one-hot encoding of the message emitted by sender i. o_t is provided as the input to the neural network, and outputs are Q-values for each of the four possible navigation actions (up, down, left and right). An action is selected using an ε-greedy policy, and the temporal difference (Sutton & Barto, 1987) error is used to compute the learning loss, i.e.: $\mathcal{L}_r = \left(R_t + \gamma \max_a Q(o_{t+1}, a; \theta_r) - Q(o_t, a_t; \theta_r)\right)^2$, when $0 < t \leq T$ and 0 at $t = 0$.

In order to additionally incentivize the agents to adopt efficient behaviours, such as reaching the goal state using the shortest possible path, the random termination probability is set as $p_{term} = 1 - \gamma$, where γ is the discount factor in the Q-

[2] For the sender, we interchangeably use the terms "message" and "message action".

[3] All gridworlds used in this paper are 5×5.

Table 1: Hyperparameters used to train different sender-receiver agent configurations.

Name	Values	Description
M	$[1, 2, 3, 4, 5]$	Number of sender agents
C	$[3, 4, 5, 8, 9, 16, 25, 27, 32, 36, 64]$	Communication channel capacity (N^M)
η	[5e-5, 1e-4, 5e-4, 1e-3]	Learning rate for RMSprop
ε_s	$[0.01, 0.05, 0.1, 0.15]$	Sender's action exploration rate
ε_r	$[0.01, 0.05, 0.1, 0.15]$	Receiver's action exploration rate
γ	$[0.7, 0.8, 0.9]$	Receiver's Q-learning discount factor
layout	[Pong, Four room, Two room, Flower, Empty room]	Environments (see Fig. 2)

learning algorithm. The total loss $\mathcal{L}_{total} = \Sigma_i \mathcal{L}_{s_i}(\theta_{s_i}) + \mathcal{L}_r(\theta_r)$ is then minimized using the RMSProp optimizer with a mini-batch size 10, implemented in TensorFlow.

Training Each experiment consisting of a sender-receiver agent setup is trained for 20 million steps in a gridworld environment. Table 1 lists hyperparameter values and descriptions across experiments. In experiments that contain more than one sender, all senders have the same ε value, and the same vocabulary size N. The number of available messages depends on the number of senders, and is selected so as to allow a fair comparison among different configurations in terms of the total number of messages (more details are provided in the following sections). In the analyses, we select the runs with the best performing learning rates and exploration rates, resulting in approximately 180k experiments.

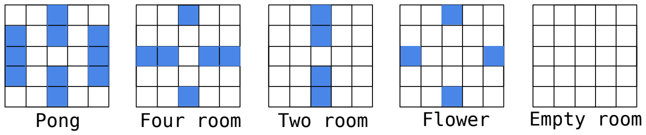

Figure 2: The five different gridworld environments used in experiments. Blue cells represent walls.

Results

To evaluate learning success, we first analyze the average returns obtained during training. The upper panel in Figure 3 shows the returns sampled at regular intervals during training for different sender-receiver configurations, for the first 12 million steps. The highest possible return for each environment is indicated by a gray line, and is calculated as the average of all possible discounted rewards in the given environment, when using the optimal policy (i.e. the shortest path). The figure also shows comparisons with two additional baselines: a single, non-communicating Q-learning agent that sees the goal, as well as a random baseline, which consists of a sender emitting random messages. In the latter case, the receiver's optimal strategy is to treat the messages as noise, and learn to visit every possible location in the environment in search for the goal.

Based on the training curves, we make two major observations. First, communicating agents are able to successfully solve this task, as shown by the learning curve approaching the theoretical maximum value. The performance of communicating agents is comparable to that of a single Q-learning agent. Second, we observe that different combinations of communicating agents, and in particular the configuration consisting of a single sender and a single receiver, often display faster convergence than the single Q-learning agent. This is apparent in all environments, except in the *Empty room* environment where they are comparable. We note that this could also be due to the lack of extensive tuning of neural network hyperparameters, such as the number of units in the hidden layer or the regularization factors. From our experiments, it appears that the single Q-learning agent was more affected by the lack of hyperparameter tuning, but as our goal was not to examine the conditions under which the agents perform optimally, we did not pursue the investigation of those differences further.

Communication channel capacity

Next, we study how the capacity of the communication channel affects the agents' performance on the task. The capacity C of the communication channel is defined as the total number of messages used in an agent setup, and is computed as $C = N^M$. For example, $C = 16$ corresponds to the following setups: 1 sender with 16 messages, 2 senders with 4 messages each, or 4 senders with 2 messages each. We examine the relationship between the channel capacity and the average return normalized by the theoretical maximum.

We know that the agents should perform well on the task if the size of the communication channel is the same as the number of non-occupied locations in the environment. In this case, every message can be used to uniquely identify one goal location. However, we are particularly interested in understanding the meaning of messages when the size of the channel is small compared to the number of possible goal locations. For example, if there are only 3 or 4 messages available, the sender needs to compress information about goal locations and use a single message to convey information about several goal locations.

The lower panel in Figure 3 shows normalized returns for different sizes of communication channel, averaged over all

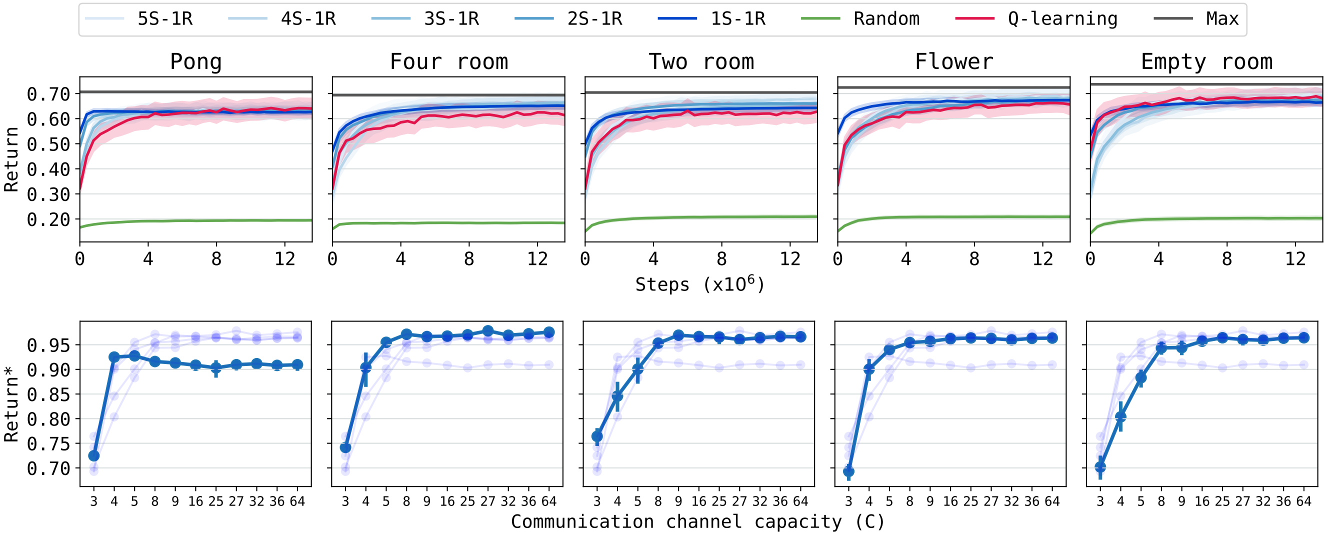

Figure 3: Upper panel: Mean training return curves for different sender-receiver agent configurations with baseline comparisons (Q-learning, Random and Theoretical maximum). Lower panel: Increase in average return depending on the channel capacity, defined as the total number of messages N^M. Abbreviations: S=Sender(s), R=Receiver. Return*=Return normalized by theoretical maximum.

agent setups. A single point on a curve is an average return by all sender-receiver agent configurations that use the total number of messages indicated by the corresponding channel capacity C on the x-axis. In all environments, we observe two distinct curve regions: a linear increase in the performance with every message added to the communication channel, and a plateau where adding more messages does not improve performance. These regions are most apparent in the *Empty room* and *Two room* environments, where we see an increase in performance up to approximately 8 or 9 messages. For the *Pong* environment, the peak is reached at about 4 messages, and the performance drops when there are more than 5 messages available.

The point after which we observe little to no improvement is different for each environment, and it approximately corresponds to the smallest number of messages needed to encode shortest paths to all possible goal locations. For example, in the *Pong* environment 4 such paths exist, so when the goal is located anywhere on one of the paths, the agent is guaranteed to reach the goal using the fewest steps possible, if it knows which path to take. For all other environments, there are 8 such paths. Such path configurations may not be unique, as for some environments there are multiple shortest paths to each goal location. While having 8 messages requires agents to compress information about goal locations, this compression still allows them to optimally solve the task, and can thus be seen as a form of lossless compression.

Subcapacity and supracapacity regime The subcapacity regime is defined as 5 or fewer messages in total for *Pong*, and 9 or fewer messages for all other environments. Approximately, these thresholds correspond to the channel capacity regions in which agents need to use lossy compression (subcapacity) or not (supracapacity). In the subcapacity regime, increasing the capacity of the communication channel by adding an additional message strongly correlates ($r > .65, p < .001$) with the improvement in performance for all environments.

We also investigate the relationship between the environment structure and the normalized performance. Here, we consider structure as a measure of the uniqueness of the optimal policies, and quantify it as the inverse of the total number of shortest paths to each location in the environment. According to that definition, we obtain the following values: 1/14 for *Pong*, 1/22 for *Four room*, 1/32 for *Two room*, 1/44 for *Flower* and 1/64 for *Empty room*. Consequently, *Pong* is the environment with most structure, and *Empty room* has the least structure. This measure is also related to the amount of empty space in the environment, as environments with more empty space are less structured.

We find that agent pairs in the subcapacity regime, specifically in the low regime of $C = 3,4$ achieve higher normalized return in environments with more structure ($r = 0.42$, $p < .001$). In this case, the structure is helpful insofar as obstacles in the environment restrict possible paths for the receiver. This effect is reduced as more messages are added, and even reverses in the supracapacity regime.

Emergent Communication Protocol

We will now focus on the analysis of the agents' policies in order to characterize the learned communication protocol. In order to understand the information conveyed by the senders' messages, we manipulate goal locations, and examine what effect the manipulation has on the emitted messages. The

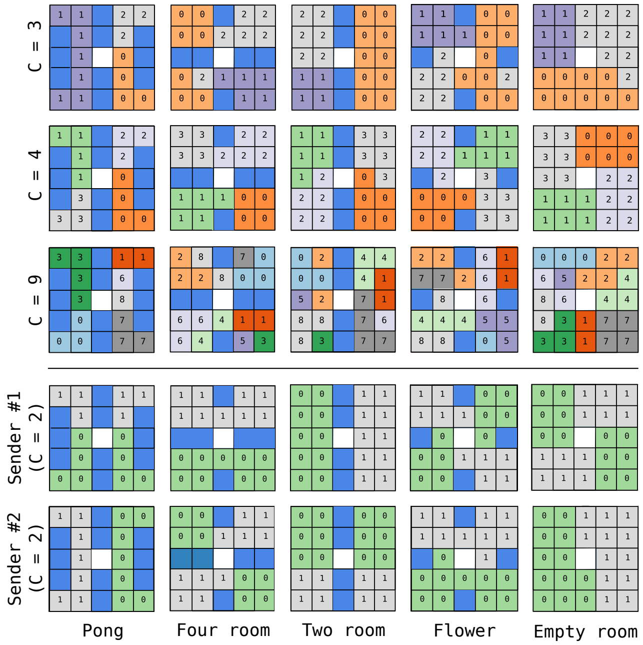

Figure 4: Sender's policy: Learned message distributions for goal locations for different channel capacity sizes (C) and different agent setups (upper panel: 1S-1R, lower panel: 2S-1R).

manipulation consists of placing the goal at all possible locations in the environment. We also analyze the receiver's policy by manipulating messages and observing the resulting trajectories in the environment.

In this section we will be predominantly concerned with the following two questions:

1. What message does the sender choose for goal *(x, y)*?
2. Where does a receiver go if it receives a message *m* while at location *(x, y)*?

Qualitative analysis To answer the first question, we follow the procedure shown in Figure 1A). From a trained sender-receiver model we use the sender's network, and probe it with a one-hot encoded vector that represents a single goal location *(x, y)*. We then use greedy action selection ($\varepsilon = 0$) to select a message at the output. This process is repeated for all goal locations for a specific environment, obtaining a single message for each location. Then, locations with the same message are color-coded, yielding a single plot in Figure 4.

The upper panel contains message distributions from 15 different experiments, each one corresponding to a different 1S-1R setup in different environments and for channel capacity sizes of $C = 3, 4, 9$. The center, corresponding to the location $(2, 2)$, is left blank in all plots, as the starting position of the receiver can never be a goal location.

Message distributions generally produce clustered regions in space, so that goal locations anywhere in a region are described by a single message. The effects of lossy compression are apparent when $C = 3, 4$, as in such cases a single message is assigned to multiple goal locations. Such regions are also interpretable: for example, when $C = 3$ in the *Pong* environment, a single message encodes *left of the center*, and in the *Two room* environment, a single message encodes *right of the center*. Finer spatial interpretations emerge with $C = 4$, where message clusters encode regions that can be described as individual rooms, such as those in *Four room*, or equal and symmetrical areas of space, as those in *Flower* ($C = 4$) or *Empty room* ($C = 4$).

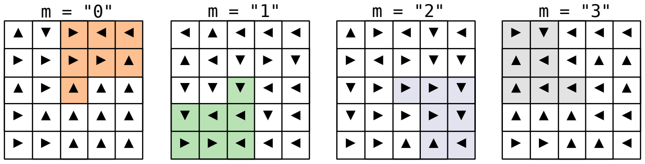

Figure 5: Receiver's policy: Navigation trajectories as actions with the highest Q-value for each location and each message *m* for *Empty room* ($C = 4$).

With this kind of compression taking place, we often observe that the receiver adopts a "sweeping" goal search strategy. Figure 5 shows the receiver's policy for the sender's policy *Empty room* ($C = 4$) shown in Figure 4. For each possible message, the receiver's greedy actions for each location *(x, y)* are depicted as arrows. That is, a single arrow at a location provides an answer to the question 2 above. Iteratively answering this question yields a trajectory through the space, starting from the center.

In this example, all trajectories cover every location in their highlighted region of space only once, before looping back to a previously visited location. Assuming the goal is located on the trajectory, the path defined by that trajectory is Hamiltonian and is thus an optimal navigation strategy for this capacity regime.

As we increase the size of the capacity channel, we observe that messages start encoding shortest paths to each possible goal location, as seen for $C = 9$ in Figure 4. Adding messages shortens the average path length that the receiver takes to reach the goal, which explains why we see such improvements in the subcapacity regime in Figure 3. In most cases in Figure 4 ($C = 9$), a single message is used to signal goal locations lying on the same shortest path, but we also observe cases where a single message is used to encode a single goal location (e.g, "7" and "3" in *Four room*, or "3", "5" and "6" in *Two room*). In general, as we increase the size of the communication channel we observe a preference in the sender for assigning a single message to a single location.

Multisender agent setup Message distributions for selected agent setups consisting of two senders, with two available messages each, are shown in the lower panel in Figure 4. The panel shows results from five different experiments; a single experiment consists of a (Sender #1, Sender #2) plot.

We observe a highly coordinated allocation of messages, so that if the first sender allocates the messages by partitioning the space according to one axis, the second sender does the same with a different, possibly orthogonal axis. For example, in the *Pong* case, Sender #1 partitions the space according to the y-axis, using the message "0" to denote "down" and "1" to denote "up". Sender #2 then partitions the space according to the x-axis, by assigning message "1" to all goal locations to the left of the center, and "0" to all goal locations to the right of the center. The senders learned to coordinate their actions even though they are individual agents that only share the information about the reward r.

If we consider multiple senders as a single sender emitting several independent messages, we can observe a basic form of compositional structure (Fodor & Pylyshyn, 1988). Thus, an important feature of formal and natural languages (Frege, 1892) emerges based on minimal assumptions about interactions between individual messages.[4]

Lastly, we evaluate the relative impact of individual senders on the task performance. We investigate whether messages from all senders contribute equally to the receiver's ability to solve the task. If all messages are equally important, we expect to see an equal drop in task performance when we scramble each message individually. We test this hypothesis by letting trained agents from the 5S-1R setup perform the task for 1000 episodes (with $\varepsilon = 0$ for both agents), with the following modification: we replace a single message in the sequence of 5 messages with a random message from that sender's vocabulary. This is done iteratively for each message in a single experiment, and we calculate the average return after each episode. Thus, we get five returns for one 5S-1R setup that we sort in descending order. We use those returns to compute the drop in performance relative to the baseline, which is the performance of trained agents on 1000 episodes without any modification to the communication channel.

The results in Figure 6 show the average drop in performance for all 5S-1R setups. Since we observe a gradual decrease in performance drop with each subsequent sender, we can reject the hypothesis that each sender contributes equally. Scrambling the message from a single sender can cause a performance drop anywhere from 28% to 88%. Thus, while all senders are important for achieving the best performance on the task, some senders contribute more than the others.

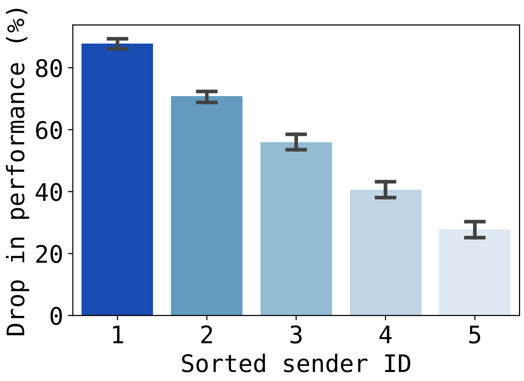

Figure 6: Average drop in task performance caused by scrambling each sender's message in 5S-1R setups. 95% bootstrapped confidence intervals are shown.

Conclusion and Future Work

Emergent communication in multi-agent reinforcement learning is a promising avenue for creating AI systems that can learn adaptive communication strategies. We have shown that situated agents are able to learn an interpretable, grounded communication protocol that allows them to efficiently, and in many cases, optimally, solve navigation tasks in various gridworld environments. By scrutinizing the agents' policies, we have observed that signals such as *left*, *up*, or *upper left room* emerge, in such a way that the state space is clustered spatially with minimal inductive bias. Using populations of agents to obtain a sequential representation of signals, we have shown that the learned protocol exhibits basic compositional structure as well as signal dominance. Future work will examine those properties and their relationship to natural language in more detail.

Acknowledgments

The authors would like to thank Angeliki Lazaridou for providing useful advice that helped set the direction of the project, Thomas Colin and Terry Stewart for the detailed commentary on the manuscript. In addition, they are grateful to their colleagues at DeepMind and MILA for many useful discussions.

References

Bard, N., Foerster, J. N., Chandar, S., Burch, N., Lanctot, M., Song, H. F., … others (2020). The hanabi challenge: A new frontier for ai research. *Artificial Intelligence*, *280*.

Bohn, M., Kachel, G., & Tomasello, M. (2019). Young children spontaneously recreate core properties of language in a new modality. *PNAS*, *116*(51), 26072–26077.

Clark, H. H. (1996). *Using language*. Cambridge University Press.

Fodor, J. A., & Pylyshyn, Z. W. (1988). Connectionism and cognitive architecture: A critical analysis. *Cognition*, *28*(1-2), 3–71.

Frege, G. (1892). Über Sinn und Bedeutung. *Zeitschrift für Philosophie und philosophische Kritik*, *100*, 25–50.

Kottur, S., Moura, J. M. F., Lee, S., & Batra, D. (2017). Natural language does not emerge 'naturally' in multi-agent dialog. In *EMNLP* (p. 2962-2967).

Lazaridou, A., Hermann, K. M., Tuyls, K., & Clark, S. (2018). Emergence of Linguistic Communication from Referential Games with Symbolic and Pixel Input. *ICLR*.

Lewis, D. (1969). *Convention: A philosophical study*. Harvard University Press.

[4] The expressivity of this protocol is bounded by limiting the number of concatenated messages in a single expression, thus impacting its productivity.

Li, F., & Bowling, M. (2019). Ease-of-teaching and language structure from emergent communication. In *NeurIPS* (pp. 15825–15835). Curran Associates, Inc.

Lowe, R., Foerster, J., Boureau, Y.-L., Pineau, J., & Dauphin, Y. (2019). On the pitfalls of measuring emergent communication. In *AAMAS Proceedings* (pp. 693–701).

Sutton, R. S., & Barto, A. G. (1987). A temporal-difference model of classical conditioning. In *CogSci Proceedings* (pp. 355–378).

Watkins, C. J., & Dayan, P. (1992). Q-learning. *Machine learning*, *8*(3-4), 279–292.